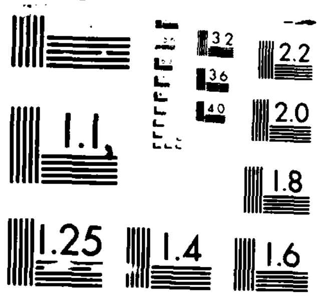
AD-A179 514 AIRCRAFT READINESS UNDER BUDGET CONSTRAINT ITS RELATIONSHIP WITH LOGISTIC..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R M LOPES 1/2

UNCLASSIFIED DEC 86 AFIT/GOR/OS/86D-8 F/G 1/3 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS

AFIT/GOR/OS/86D

AIRCRAFT READINESS UNDER BUDGET CONSTRAINT,
ITS RELATIONSHIP WITH LOGISTICS RESOURCES;
THE SPARE PARTS COMPONENT,
A MODEL FOR THE PORTUGUESE AIR FORCE.

THESIS

Rui M. Pereira Lopes
Major, Portuguese Air Force

AFIT/GOR/OS/86D-8

AFIT/GOR/OS/86D-8

AIRCRAFT READINESS UNDER BUDGET CONSTRAINT, ITS RELATIONSHIP WITH LOGISTICS RESOURCES; THE SPARE PARTS COMPONENT, A MODEL FOR THE PORTUGUESE AIR FORCE.

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fullfillment of the
Requirements for the Degree of
Master of Science in Operations Research

Rui Manuel Pereira Lopes
Major, Portuguese Air Force

December 1986

Preface

Recent cuts inflicted in the budget of the Portuguese Air Force directed the attention of the decision makers to the need of relate aircraft readiness with logistics resources. The purpose of this research was to provide the Portuguese Air Force Logistics Command managers with a model which relates aircraft availability with expenditures on spare parts needed to carry on the inventory. The optimal allocation of funds to spare parts acquisition is also provided by the model.

Few research projects are the results of one man's effort. This is particularly true of this one. I am indebted to Major Ronald Stokes and Fred Rexroad of AFLC/XRSC for furnishing all documents related with the AAM model and giving the necessary assistance, explanations and guidance to the understand of the model. I would especially like to thank my thesis advisor, Major Joseph R. Litko. His advice and assistance has been invaluable.

Extending to the Portuguese Air Force, I would like to thank Major Patricio, Captain Domingos and Captain Jesus for their contribution with the tremendous effort on the data collection in order to test the model. To the Portuguese Liaison Officer at AFLC, Colonel Martins and his family, I have a special thanks for their continuous moral support and encouragement along all these months.

Finally, a very special thanks to my wife, Isabel, and my son, Joao Ricardo, for their patience and understanding during this long period of separation.

Rui Manuel Pereira Lopes

Table of Contents

Page

List of Figures

List of Tables

AFIT/GOR/OS/86D-8

Abstract

One of the general concerns of the Portuguese Air Force top managers is to relate resources to readiness. This research addresses the relationship between spare parts and aircraft availability, as a component of the general problem of relating resources to readiness. As background, the theoretical development and problem solution techniques of METRIC: A Multi-Echelon Technique for Recoverable Item Control, MOD-METRIC: A Model for Multi-Item, Multi-Echelon, Multi-Indenture Inventory System, and AAM (Aircraft Availability Model) models are presented. After identifying the major mathematical issues and contributions of each model to the solution of the problem, an easy-to-use mathematical computer model, that simplifies the actual Aircraft Availability Model in use by the Air Force Logistics Command (~~AFLG)~~ and that fits the requirements of the Portuguese Air Force, is presented. The simplified model for aircraft availability provides solutions that are very close to historic values. An additional benefit of the simplified model is that it can be used to predict either aircraft availability or total expenditures. It is recommended that weapon system managers in the Portuguese Air Force use the model to budget the spare parts requirements for each aircraft type.

AIRCRAFT READINESS UNDER BUDGET CONSTRAINT,

ITS RELATIONSHIP WITH LOGISTICS RESOURCES;

THE SPARE PARTS COMPONENT,

A MODEL FOR THE PORTUGUESE AIR FORCE.

I. Introduction

General Concern

One of the principal concerns of the Portuguese Air Force Logistics Command is the need to relate resources to readiness. Readiness is defined as the ability of weapon systems to perform their assigned missions. To measure readiness, the interactions between hardware reliability, the operational environment, and the logistics system must be known. For this reason, the percentage of scheduled flying hours actually flown during a specific time period is used as the measure of readiness.

Throughout the last two or three years it has been difficult to accomplish the flying hour goal, partly due to budget restrictions. Although the budget is very restrictive, it is suspected that other factors may significantly influence the accomplishment of this goal. The identification and analysis of these factors are very important for more precise and adequate determination of the level of readiness.

To measure readiness, as stated before, it is necessary to quantify the interactions between three areas: hardware reliability, operational environment and the logistics system. The hardware reliability is projected on the frequency of maintenance actions needed to keep the system operating. The performance of logistics systems is measured by the frequency of maintenance actions, the support equipment necessary to accomplish them, and the level of spare parts available when supporting the operational requirements. Thus, a weapon system is ready to accomplish its mission when all required maintenance actions are performed and no spare parts are missing.

Problem Statement

In the past two or three years, the Portuguese Air Force has come to realize that it needs more precise data to better grasp the implications of its decisions on the funding of weapon systems effectiveness-related accounts. Resource constraints and funding shortfalls have generated a wave of concern about the readiness of the Air Force. As part of the readiness problem, the interaction between spare parts budget allocation and aircraft availability is not adequately understood by Portuguese Air Force managers. Concomitant with this problem there is another one. Portuguese Air Force management has not developed adequate tools to measure or define the interrelationships between them.

Brief Background

A previous study done at Portuguese Air Force headquarters (23) addressed the general concern of relating resources to readiness. The result was a model which uses available data from the last three years as its input. The actual readiness rate attained during this three year period was included as part of the input. However, the model is inadequate in that the interaction between the budget constraint, level of spare parts and aircraft availability was not addressed.

As a result of the strong budget constraint, the levels of spare parts have been reduced to a point that adversely affects weapon system availability. For a better utilization of scarce resources it is necessary to have a responsive link between the budget allocated to spare parts and weapon system availability.

In any spare parts inventory, a large part of the investment is concentrated in assets referred to as recoverable items; that is, items that can be repaired and restored when they fail (10:472). Because of their importance the U. S. Air Force has developed at least three inventory models to help determine the proper recoverable item stock levels for a given investment in spare parts. These three models, METRIC: A Multi-Echelon Technique for Recoverable Item Control, MOD-METRIC: A Model for Multi-Item, Multi-Echelon, Multi-indenture Inventory System, and the AAM (Aircraft Availability Model), are currently used by the Air Force Logistics Command (AFLC). They will be the baseline in this research effort. A more detailed overview of them will be presented in Chapter II.

METRIC Model. METRIC models a two-echelon (base/depot), multi-item inventory system (20). Its purpose is to determine the proper stock levels for recoverable items which are part of a larger end item (for the purpose of this research it is considered the aircraft). One of its drawbacks is that only recoverable parts at the first level can be modeled; that is, the recoverables that are directly applied to the aircraft as an item.

MOD-METRIC Model. MOD-METRIC models the more complex environment of a two-echelon, two-indenture inventory system (10). By two-indentures we mean the components are modularly designed; that is, the recoverable items themselves contain recoverable items. The purpose of this modular design is to shorten the repair time of the primary components (by simply removing and replacing the recoverable modules).

The current implementation of MOD-METRIC on the AFLC CREATE computer system (2) presents some difficulties when a recoverable item being modeled contains many modules and/or is located at many bases. One problem is the computation time associated with one run of the program; another problem is the limited range of solution points (total budget). To find each solution point the approach taken is to partition the problem into an LRU subproblem and an SRU subproblem. Each subproblem is then solved for several different trial budgets subject to a budget constraint. The selection of the budget constraints for the subproblems is arbitrary. However, the subproblems together must be subject to a total budget constraint. It is clear that the solution depends on the bounds selected for the LRU and SRU investments.

AAM. AAM is a generalization of the MOD-METRIC problem with a different objective function (11). While using METRIC concepts to model a multi-indenture inventory system, marginal analysis techniques are employed to relate the recoverable items budget with the aircraft availability rates.

Literature Review

This research effort is concerned with determining aircraft operational effectiveness based on the maintenance and supply support characteristics. A system's effectiveness is a multi-faceted characteristic. Sherif and Kheir have defined it as "... the probability that the system can successfully meet an operational demand within a given time when operated under specific conditions" (22:1). One of the measures of this quality is readiness, which of itself can be defined any number of ways. For example, and using the same source, readiness is define as "... the probability that at any point in time [the system] is either operating satisfactorily or is ready to be placed in operation on demand when used under stated conditions" (22:5). Others, including R. H. Monahan and the Department of Defense Readiness Management Steering Group, have defined it more generally as the ability of the system to perform the mission for which it was designed during a normal operating cycle (8:3;9:3; 14:C-11). Readiness is not easily defined. In fact, Monahan (9:12-16) compiled a list of sample definitions used within the Department of Defense. Each of these definitions is related with the scope of functional requirements of readiness. They are specific at the bottom level of the echelon hierarchy, and broader at the top level. Echelon dependency is one factor inherent in readiness definition. Thus, in this research, readiness is defined as the probability that a system mission is launched when it is scheduled. In other words this means that all maintenance actions must be performed and the required spare parts must be available in order to have the aircraft ready to accomplish its assigned mission.

In the Aircraft Readiness Analysis Program of the Portuguese Air Force (16:2-8), aircraft readiness (AR) is defined as the ratio between Mission Capable (MC) aircraft and the number of Aircraft Available (AA) at each base.

$$AR = MC/AA$$

In order to evaluate the contribution of spare parts to the problem of aircraft readiness, the aircraft availability concept must be introduced. An aircraft is considered to be ready, or MC, when it is not waiting for maintenance actions to be performed on it and all spare parts needed for the mission are available. Thus, the measure of aircraft readiness or Mission Capable is a function of two other measures; one for the maintenance system, Not Mission Capable due to Maintenance (NMCM), and another for the supply system, Not Mission Capable due to Supply (NMCS). Aircraft availability rate is the quantity used to measure the performance of the supply system. Similarly, an aircraft is defined to be available if it is not missing a part. The availability rate for an aircraft type is then the percentage of aircraft available over a specified time period.

The literature review for this research effort is divided into two phases involving the subject of readiness. Phase 1 consists of a review of the readiness-analysis problem, or more specifically, a subset of the overall problem, namely, the logistics resources-to-readiness issue. Phase 2 consists of an investigation on the current status of research in what concerns the logistics-readiness discussion, or more specifically, on money allocation among competing resources, namely, spare part stocks.

Readiness-Analysis Problem. The subject of readiness has been extremely prevalent not only in the Portuguese Air Force but also in the United States Department of Defense. Much time and money have been spent studying the readiness-analysis problem, including defining readiness, measure readiness, relating logistics resources to readiness, and evaluating the importance of readiness.

While all of these facets of the readiness issue are important, there has been, in the past two years, an emphasis on the third element of the problem. However, recognition of the need to relate resources to readiness is not new. The following statement concerning logistics problems is attributed to Vice Admiral Gaddis, USN, in 1974.

> An example of logistics problem is our need for a simple, usable definition of material readiness of Naval forces, a means of measuring it, and some perfectly definite input-output relationships. We need to be able to link resources inputs, and this means money, to any of the numerous potential outputs, and these mean military applications. We need to be able to predict not only how much readiness measure will change, but also when it will change, as a result of changes in inputs. (9:11)

Although Vice Admiral Gaddis was concerned with the Department of the Navy, his statement is equally applicable to the Air Force.

The Congress took actions to increase attention to relationships between resources and readiness in the 1978 Defense Authorization Act. This Act required that the Department of Defense budget submission to the Congress "...

include data projecting the effect (on readiness) of the appropriations requested for material readiness requirements" (7:1). The Department was also to submit to the Congress "... a report setting forth quantifiable and measurable material readiness requirements" (7:1).

When attempting to comply with these requirements, the Office of the Secretary of Defense faced a number of problems. First, there were no clearly defined or agreed-upon measurable material readiness requirements. Secondly, some goals existed, but they were generally not related to any analysis of combat capability. Thirdly, there was no ability to project the effect of appropriations on materiel readiness (7:1; 1:15; 8:3).

Because of this and other weapon system acquisition problems, Deputy Secretary of Defense Frank Carlucci chartered five working groups to make recommendations to improve the acquisition process used by the services, and early in 1981, he identified a list of actions for implementation by the Department of Defense. A number of these actions involve the readiness issue. They include steps to improve system support and readiness, to improve reliability and support, and to increase program manager control over support resources (4:54).

To implement these Defense Acquisition Initiatives requires methods of relating resources to readiness. This led to the current emphasis on developing readiness models which consider the logistics support elements. The normal approach to this modeling problem has been to use detailed models of the system's operational and maintenance environment. An example of application of this modeling technique of detailed environments is the LCOM model. There are, however, at least two distinct disadvantages to this approach. First, these types of models generally require a large amount of detailed data which probably is not available early in the system life cycle when the program manager must make important logistics planning decisions. Secondly, it is not uncommon to require several months to a year to set up the networks and data files for a run of a model such as the MOD-METRIC model or the Aircraft Availability model (AAM). For additional information on these models, the reader is referred to their User's Manuals (12 and 5, respectively). These large simulation models often require several hours to run in batch mode. Generally, they can not be run in interactive mode. These disadvantages limit the number of "what-ifs" which the analyst can consider as well as the responsiveness to those few options which the program does want to be investigated. A simplified version of the AAM can be developed in order to be an easy-to-use model which will run quickly with a minimum amount of data. This model will have the necessary requirements to relate aircraft availability to spare parts levels, minimizing the investment. This model is, of course, not the only one in existence that attempts to provide the capability to perform this type of analysis. However, it gives the necessary insight in order to apply its concepts to the Portuguese Air Force needs.

Logistics-Readiness Discussion. With the downward movement of the defense budget, the Portuguese Air Force is experiencing a period of underfunding for logistics resources. One of the most significant problem areas involved the funding of replenishment spare parts. This logistics resource category includes all of the higher cost assemblies and subassemblies required to keep individual aircraft operationally ready. As these items fail, they must be repaired at base or depot level; or if not economically reparable, they are condemned and discarded. Stocks of these items must be maintained at base or depot level to replace failed items that are condemned or placed in the repair cycle.

Because of the current lack of sufficient funding, these stocks were drawn to low levels. This resulted in decreased numbers of mission capable aircraft.

To some extent, this erosion of the optimal base number of spares took place due to the inability of the Portuguese Air Force management to directly relate funding levels and aircraft readiness. Readiness is measured by the availability of spare parts and the accomplishment of the maintenance actions required to keep the aircraft ready for the mission. The availability of spares is a function of the level carried on the inventory which, in turn, is a function of the demand (failure rate) and unit price. Thus, the purchase price should be one of the elements to take into consideration when determining optimal amounts of spares for the corrective maintenance. Petrovic, Senborn and Vujosevic have developed a combined optimization/simulation approach to determine the spares needed in an inventory when some of the parameters are not completely known. The system components are classified into subcategories, defined by the range of failure rate and its representative unit price. The subcategories are optimized until the performance measure of spares is satisfied (15). This suggests the use of marginal analysis techniques in selecting the greatest improvement towards the optimal solution. In particular when selecting from ranked elements or categories.

The lack of funds can not be claimed as the only problem found when relating resources to readiness. Misallocation of money among different resources is another problem. Given a certain distribution of resources and a certain amount of money for replenishment, the optimal use of the money for replenishment must be found. This problem can be approached in different ways. For example, given a different number of items to repair or to buy and a certain amount of money to invest in the required resources, find the optimal equivalent in resources of the available money. Several authors addressed this general problem of allocation of resources. Einbu has presented an algorithm for the assignment of resources to depots and for establishment and replenishment of depot stockpiles (5). The problem examined is that of distributing resources among a set of locations (depots) so as to optimize an objective function associated with a given level of activities.

The interface between logistics resources and readiness is not complete without addressing the importance of maintenance manpower. Readiness, when measured as the availability of the aircraft over a certain period of time, can be evaluated by maintenance manpower requirements. As described by Blom and Evans, "Logistics manpower models ... establish the relationship between Air Force program variables and logistics workload requirements" (3:18). Flying hour and inventory factors are developed for these logistics manpower models. Using these factors, it is possible to project manpower requirements to accommodate changes to the inventory or number of flying hours. With this type of relationship, the financial resource allocation required for each flying hour can be identified.

<u>Summary</u>. The literature research consisted of two main areas, readiness-analysis and logistics-readiness. In the first area, published literature by the Department of Defense was selected to show its concern with the subject of

readiness. The current development in readiness models, which consider the logistics support elements were analyzed. METRIC, MOD-METRIC and AAM are three of these models. Two main disadvantages in these models were identified. One of them is the need for large amounts of data. The other is the long data file set up time required due to the level of detail of these models. Because of these disadvantages, the needs of the analysts during the "what if" analysis are not met. Therefore, a quick-answer model is required for preliminary research which might lead to detailed analysis of promising options. A simplified version of the AAM has the capability to relate aircraft availability to spare parts levels minimizing the necessary investment.

In conclusion, this literature review gives the essential insight required to identify the concepts for a model development that can meet the needs of the Portuguese Air Force. With this model, the budget allocation to spare parts problem will be solved with the total flying hours goal maximized for a certain level of funding.

Scope and Limitations

As stated before, the concern of this research is related with identification and analysis of the interaction between spare parts and aircraft availability, a component of the problem of relating readiness to logistics resources. Logistics resources must be supported by the Portuguese Air

Force budget. Every year, the Air Force is faced with the same problem: there is a conflict between achieving operational requirements and allocating sc .ce financial resources for maintenance and supply. The main concern of this report is the allocation of the budget to spare parts in order to maximize the number of aircraft available, giving the opportunity to maximize the operational flying hours.

There are a number of constraints on the research effort. These constraints are needed as an attempt to bound the research within the time frame available. The scope of the study is defined by the spare parts component of the logistics resources. Since almost all of the budget allocated to spare parts is spent on recoverables, the majority of the attention in this research will be given to these components. Only one type of aircraft is considered in developing the model. As a follow-on in this effort other aircraft can be added to compete for these resources. Also, the modeling of manpower and support equipment requirements can be the subject of further research.

Research Objectives

It is the purpose of this research to develop an easy-to-use computer model, which runs quickly and requires a minimum amount of input data. Spare parts is the component of logistics resources which will be specifically considered by this model. It will use this resource to determine the availability of a specific aircraft type given a certain level of budget and the total flying hour program established. Also, the model will allocate the budget in order to maximize the level of aircraft availability.

Research Questions

In order to meet the research objectives, the attempt to answer the following questions will give the correct directions to be followed.

1. What are the relationships between spare parts budget levels and aircraft availability?
2. Can a conceptualization of the interrelationships between aircraft availability and the spare parts needed to support the flying hour programs be developed and used as the basis for an easy-to-use mathematical computer model?
3. Can the developed model function as a management tool, whereby, managers can determine the effect of proposed changes in spare parts budget on aircraft availability ?; or,

   Can they, given an aircraft availability goal, determine the budget level required for spare parts ?

Overview

In the following chapter, Chapter II, the supply system and the repair cycle concepts will be described. Included in this chapter an overview of the METRIC, MOD-METRIC and AAM models will be presented with more detail. The chapter will be finished with the statement of the assumptions and the description of the context in which the model to be developed will be used.

Supported by the theory, concepts and assumptions of the METRIC, MOD-METRIC and AAM models, chapter III presents the methodology used in developing the two-echelon, two-indenture model for one aircraft type. The implementation of this model in the Portuguese Air Force is also discussed.

Chapter IV presents the model with more detail. It begins with a narrative description and follows with a discussion on verification and validation of the model. A baseline scenario is defined and the model tested against historic data. This chapter concludes with the discussion of the results and analysis of sensitivity of the model.

In chapter V, the last chapter, the summary and conclusions of the research will be presented. Managerial implications that follows from the use of this model and suggested areas of further research are also presented.

All batch files, fortran programs and subroutines are included as appendices to this report.

II. Background:

Overview of the METRIC; MOD-METRIC and Aircraft Availability Models

This chapter describes the supply system and the repair cycle concepts used along this research, and gives an overview of METRIC; MOD-METRIC and Aircraft Availability inventory models. Each of these models has been developed to determine base and depot stock levels of several recoverable items so that a particular system performance measure is optimized for a specified total cost. The discussion of each model is organized in the following sections: system environment, assumptions, data requirements, model objective and problem solution. METRIC is more fully described by Sherbrooke (20); MOD-METRIC by Muckstadt (10), and the Aircraft Availability Model by O'Malley (11). Assumptions and description of the context in which the model to be developed will be used, are presented by the end of the chapter.

Supply System Concept

For the scope of this research the supply system is composed of one central depot with repair capability and several operational bases.

The spare parts carried in inventory by this supply system fall in two categories: consumables and recoverables. Consumables are all low-cost and high-demand items that, and in general, are not reparable. Recoverables are all high-cost and low-demand items that, when a failure occurs in operation, they can be removed and repaired. The management of these two categories of items follows two well known inventory policies. The (s,S) policy is used to manage the consumable category. This policy specifies that, when the quantity on hand plus on order minus backorders is less than or equal to the reorder point, s, an order is placed to bring the inventory position up to S. The management of high-cost and/or low-demand items follows the (s-1,s) policy. This policy means that a reorder is placed whenever a demand occurs, and this is considered the optimal policy for this class of items. Most aircraft recoverable spare parts that may cost hundreds of dollars and are typically demanded a few times a year at an individual base are included in this important class.

Repair Cycle Concept

The repair cycle is based on a two echelon system. Items that have failed are removed from the aircraft or end item and repaired at the base facility if possible. If a serviceable item is available at the base supply center, it is installed on the aircraft while the failed item is being repaired. In this situation, once the item has been repaired, it becomes part of base stock. If no replaceable item is available, the item is returned to the aircraft after being repaired.

Frequently, the base is unable to repair the failed item. These incidents are known as Not-Reparable-This-Station (NRTS). NRTS items are returned to the depot for repair or condemnation. When this action is required, a demand is placed on the depot for a serviceable item. When the depot has the item in stock, it is sent to the base prior to the arrival of the failed item at the depot. If no item is in serviceable stock at the depot, the resupply is delayed until an asset returns from the depot repair cycle.

```
                ----------           ----------
    ..........| Base     |_________| Base     |
    .         |   Repair |         |   Stock  |
    .          ----------           ----------
 ----------                            |    |
| Aircraft |--------------------------      |
|(End item)|                                |
 ----------                                 |
    .           ----------          ----------
    ..........| Depot    |_________| Depot    |
              |   Repair |         |   Stock  |
                ----------          ----------
```

Legend

... Failed component

--- Serviceable component

Figure 1. Repair cycle

Upon arrival at the depot, the unserviceable item from the base goes through the depot repair cycle and becomes a part of the depot stock. When the serviceable item is sent to the base from the depot, it is given to the maintenance facility for replacement on the aircraft or is returned to base stock. This completes the cycle for a recoverable asset. Figure 1 depicts this two echelon environment.

The fundamental decision in the two echelon environment is the determination of the total number of assets required to support the weapon system. The total requirement consists of those assets in stock at base supply, in repair at the base, in the depot repair cycle, in stock at the depot, and those in the shipping process between the base and the depot.

METRIC: A Multi-Echelon Technique for Recoverable Item Control

System Environment. METRIC introduces a new concept in computing stock levels. Instead of computing them on the basis of artificial estimates of holding cost rate and backorder cost, its approach focuses management attention on the entire weapon system so that an appropriate combination of system effectiveness and system cost can be selected. METRIC models a two-echelon system such as the one described in the previous section. The goal of METRIC is to determine both requirements and distribution of recoverable items in a two-echelon inventory system.

Data Requirements. Data items related to the repair cycle described above, in addition to other data such as cost and demand (failure) rates, are the required inputs for the METRIC model. This data is as follows:

- Number of recoverable items;
- Number of bases;
- Cost of the item;
- Expected base repair time;
- Expected order and ship time;
- Expected depot repair time;
- Probability that a failure of an item requires base repair;
- Expected number of demands for an item at a given location; and
- Stock level for an item at a given location.

Assumptions. METRIC assumes that demand for each item is described by a compound Poisson process. A compound Poisson process is a generalization of the Poisson process which allows more flexibility in describing demands, while retaining the simple analytic properties of the Poisson process. Compound Poisson processes are well known and discussed in other works (6 and 21). The logarithmic Poisson process, a member of the compound Poisson family, assumes that demands arrive at the system in batches where the number of batches per time period follows a Poisson process and the number of demands per batch has a logarithmic distribution. This is particularly useful and METRIC makes use of it.

Sherbrooke (21:11-12) showed that when demands per batch follow the logarithmic distribution and batches arrive according to a Poisson process then the resulting compound Poisson process is described by the negative binomial distribution.

Feeney and Sherbrooke highlight two important properties of the compound Poisson distributions (6:394):

> 1. Any compound Poisson distribution has a variance that exceeds or equals its mean. When the variance equals the mean the compound Poisson reduces to a simple Poisson.
>
> 2. The Compound Poisson distributions are the most general class of "memoryless" discrete distributions, i.e., the number of demands occurring in a time period does not influence the probabilities of demand in any other non-overlapping time period.

These properties are important because the demand data usually produces variances that exceed the mean. Thus, the compound Poisson distribution allows us to model a demand process with variance to mean ratio greater than one, yet still maintains the "memoryless" feature of the Poisson process.

METRIC assumes that the demand process for each item has a constant variance to mean ratio, V/R, over all bases, but V/R may vary among items.

The demand process is assumed to be stationary; that is, the arrival rate and the V/R of the compound Poisson process are constant over time. Although this implies that METRIC is primarily for use as a long range planning tool in a steady-state environment, program data (such as flying hours per time period) can be adjusted to reflect different assumptions concerning the steady-state operational levels.

It is assumed that a failure of one type of item is statistically independent of those that occur for any other type of item.

In METRIC lateral resupply between bases is ignored. When a base ships an item to the depot and requests a replacement, a serviceable item will be resupplied from the depot if available. If the depot has no item on stock, the base must wait for a unit from the depot repair cycle. By not modeling this process, METRIC provides slightly conservative stock levels.

The modeling of the repair process incorporates three important assumptions. First, the level at which repair is performed depends only on the complexity of repair, not on current workload at the base or depot. Second, the repair times of individual demands are statistically independent. Finally, METRIC assumes that there is no batching of units before repair begins: when a failed item arrives for repair it begins immediately. This assumption essentially states that we have no constraints on repair capacity.

One final assumption of METRIC is that different items may be given different "essentialities",that is, the objective function may provide different weights to the performance of different items.

<u>Model Objective</u>. The objective of this model is to determine the base and depot stock levels (Stock is defined to be the total stock on hand plus on order plus in repair minus backorders) which minimize total expected base level backorders for a specific set of items and bases subject to an investment constraint. Following Sherbrooke (20) and Feeney and Sherbrooke (6), a base backorder for an item exists any time there is an unsatisfied demand for that item at base level. A backorder day for one type of item results from one item being backordered for one day. Note that n backorder days may result from several causes; e. g., one item backordered n days or n items backordered one day. If backorder days are accumulated over an extended length of time, the daily expected number of backorders-days can be found by dividing the accumulated backorders-days by the number of days in the data period and calculating the mathematical expectation of this quantity.

The expected number of backorder days for one type of item at one base is given by the compound Poisson probability and the number of outstanding backorders.

The objective of the METRIC is to calculate expected base backorders summed over all items and all bases. This looks reasonable since it is assumed that aircraft availability only depends on base backorders. It is important to note that this model does not explicitly try to minimize depot backorders; depot backorders are implicitly related with base backorders.

Problem Solution. The solution technique first suggested for the METRIC problem by Sherbrooke (20) essentially uses a marginal allocation approach; a unit of stock is added to the item and base (or depot) that produces the maximum decrease in total expected base backorders per unit of cost. This procedure is repeated until the budget constraint is reached.

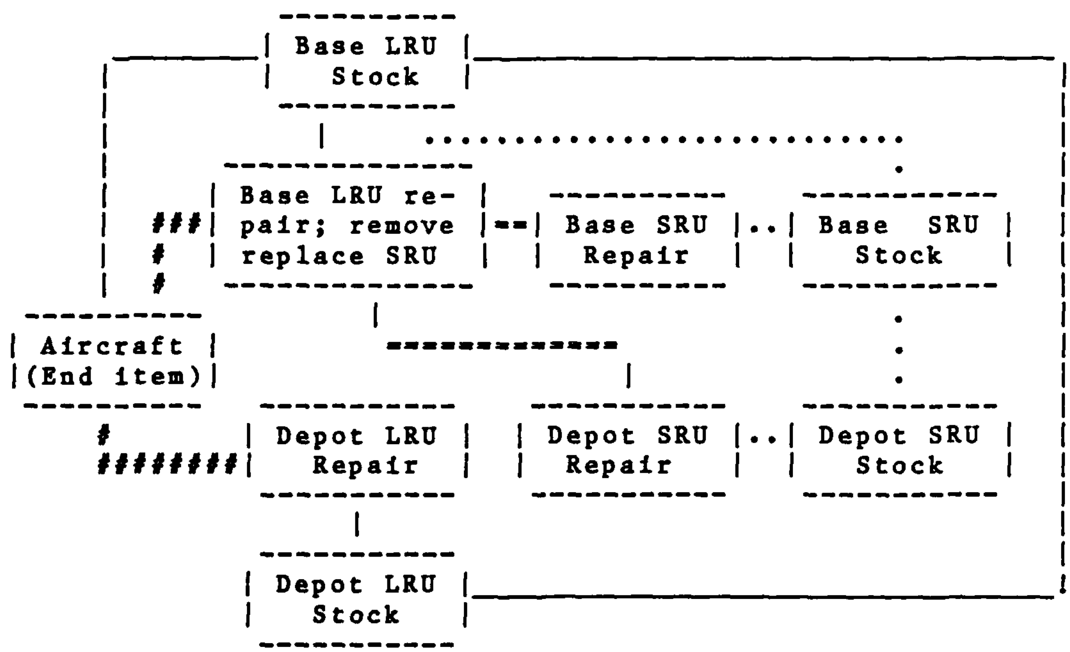
MOD-METRIC: A Two-Echelon, Two-Indenture Inventory Model

System Environment. MOD-METRIC extends METRIC to include a hierarchical or indentured parts structure. The model permits two levels of parts to be considered, an assembly and its components. Recoverable items have been redesigned to reduce repair times. The new design is based on the modularity concept. The idea is to have an assembly composed of several subassemblies or modules, that are easy to remove and replace. When the item fails it is removed from the aircraft and repaired in the shop by simply removing and replacing the failed module. The assembly is normally referred to as the Line Replaceable Unit (LRU), while the modules are referred to as Shop Replaceable Units (SRUs).

To support this concept operationally, serviceable inventories of both LRUs and SRUs are maintained at the bases and depot. When an LRU fails on an aircraft, a serviceable LRU from base stock (if available) replaces the failed LRU. The failed LRU then goes immediately to either base repair; or if necessary, is shipped to the depot and an LRU resupply request is issued to the depot. At the base repair facility, if the LRU failure has been caused by a faulty SRU, this SRU is replaced with a serviceable SRU from base stock (if available) and the repaired LRU is placed in base serviceable inventory. If no SRU stock is available, the LRU must wait for a serviceable SRU. At the depot, the LRU is assumed to be repaired (or overhauled) as an entire entity, without its SRUs being replaced.

When a faulty SRU is removed at a base, it goes through the same repair steps as the LRU (except the SRU has no modules). The SRU is a recoverable item and the (s-1,s) inventory policy is assumed. Thus, the faulty SRU immediately goes into either base repair, or an SRU depot resupply request is issued and the faulty SRU is shipped to the depot. It is the modeling of this SRU repair cycle that distinguishes MOD-METRIC from METRIC. The repair process is shown in Figure 2.

Now there are several different stock levels which will affect system performance. If base LRU stock levels are zero, an aircraft will have to wait the average LRU base resupply time before it becomes operational. As base LRU stock levels increase, the average delay in repairing an aircraft will decrease, since serviceable LRU stock will satisfy some demands immediately. Depot LRU stock levels and base SRU stock levels both affect the LRU base resupply time, but do not directly affect the average time to repair an aircraft. The goal of MOD-METRIC is to determine all these stock levels, for a single LRU and its associated SRUs, so that the total expected LRU base backorders are minimized for a specified total cost.



Legend

Failed LRU -- Serviceable LRU

== Failed SRU .. Serviceable SRU

Figure 2. LRU and SRU Repair cycle

Data Requirements. The data item requirements for the MOD-METRIC model are as follow:

- Number of SRUs associated with LRU;
- Number of bases;
- Cost of SRU;
- Cost of LRU;
- Expected base repair time of SRU;
- Expected order and ship time for SRU from base to depot;
- Expected depot repair time of SRU;
- Probability a failure of an SRU requires base repair;
- Expected number of demands (failures) for SRU at one location; and
- Stock level for an SRU at each location.

Assumptions. All but two of the assumptions under the METRIC model are valid for the MOD-METRIC model. These two, and one additional assumption are discussed below.

MOD-METRIC does not assume that the demand process for LRUs and SRUs is compound Poisson. Instead, it is assumed that only one demand is placed on the system per arrival and that these arrivals occur according to a simple Poisson process. However, the mean is assumed to be random variable described by a gamma distribution. This change of assumption in demand process is of minimal effect, though, since a Poisson demand process with a gamma distributed mean can be shown to have a negative binomial distribution. This is the same as the logarithmic Poisson process; the same distribution as in METRIC.

The other METRIC assumption not made in MOD-METRIC concerns the use of different objective function essentialities for different components. These essentialities are included in METRIC primarily to approximately model the LRU/SRU relationship that MOD-METRIC handles explicitly. That is, in METRIC formulation, an LRU and its associated SRUs would all be considered components whose backorders are to be minimized. It is clear that we would want to weight the LRU backorders more heavily than the SRU backorders; the essentialities would attempt to do this, although in a rather arbitrary way. MOD-METRIC does not need to assign these weights.

MOD-METRIC assumes that when an LRU failure is caused by SRUs, only one SRU has failed. This assumption is reasonable for most modularly designed components and greatly simplifies the mathematics in defining the objective function. It should be noted, however, that if an LRU is shipped to the depot, it may (and probably does) have multiple SRUs that need repair. Usually, when there is a shortage on spare levels, maintenance personnel stick all bad SRUs in a faulty LRU before it is shipped to the depot for repair.

Model Objective. Recall that in METRIC, the objective is to minimize expected base backorders for all items subject to an investment constraint; the MOD-METRIC objective is to minimize the expected base backorders for the end item subject to an investment constraint on the total dollars allocated to the end item and its components. MOD-METRIC only considers one LRU and its SRUs.

Problem Solution. MOD-METRIC has been implemented by AFLC (2) and it has presented some computational difficulties for problems with many SRUs and/or bases.

First, it is noted that the problem of minimizing expected LRU base backorders summed over all bases subject to a budget constraint is not separable. The approach taken, however, is to partition the problem into two sub-problems: an LRU subproblem and a SRU subproblem. Each subproblem is then solved for several different trial budgets: C(LRU) and C(SRU), where C(LRU) + C(SRU) = C and C is the total budget constraint. It is clear that the SRU subproblem must be optimized first so that the resulting SRU delay information can be incorporated into the LRU subproblem solution. After a range of trial budgets have been examined the budget division which yields the minimum objective function (in the LRU subproblem) is chosen as the optimal solution. Stock levels are found in the solution to the two subproblems for that optimal budget division.

AAM: The Aircraft Availability Model

System Environment. In almost everything the Aircraft Availability model is identical to the MOD-METRIC model. While this model models one LRU and its SRUs, AAM extends the problem to more than one LRU.

AAM considers a much more complex environment than the other models do. It models a number of aircraft stationed at several geographically distributed operational bases. Aircrafts are divided in different aircraft types, such as the F-4, B-52, or F-14. These types can be further classified into subtypes, such as F-4G, B-52H, or F-14A.

These aircraft are supported by an inventory of recoverable components stocked at each of the bases, as well as at a higher echelon of supply, or depot. The bases have a limited repair capability, while the depots have an extensive repair and overhaul capability.

The hierarchical structure of the components in the MOD-METRIC model considers two levels of indenture: LRUs and SRUs. In the AAM model this hierarchical structure continues through many levels of indenture, but the analogy with the first two levels holds. Lower indenture level backorders will be generated by demands during the repair of the component's next higher assembly (NHA) and will serve only to delay the repair of that NHA.

In practice, of course, the situation is not so simple as depicted. Sometimes a subassembly of an LRU can be removed directly from an aircraft without intervening removal of the higher assembly. Sometimes more than one SRU must be removed from an LRU during maintenance. The description presented is a simplification of actual operations but contains enough aspects of the true situation to yield an acceptable degree of accuracy, while not making the mathematical treatment intractable or the data requirements excessive.

Data Requirements. The data used by the AAM includes the standard supply data for each recoverable component, such as:

- Total daily demand rate;
- Number of using locations (number of bases to which the aircraft type using the component is deployed)
- Percentage of demands which are not repaired at base level;
- Average base repair time;
- Average depot repair time (includes retrograde time, the time to ship a failed component from base to depot); and
- Average order and ship time (from depot to base).

In addition, since an aircraft type can be composed of several subtypes the AAM requires configuration data for each aircraft type, such as:

- Quantity per application (QPA);
- Application percentage;
- Total number of aircraft; and
- Flying hour program.

Assumptions. As in the previous models, component demands are assumed to be generated by a Poisson process. The model is built around a theorem of Palm (12) which states that, for a Poisson demand process coupled with a resupply process (such as base repair), if the resupply time is independent of demand then the distribution of the number of items in resupply will be Poisson, depending only on the average resupply time and not the distribution of the resupply time. The expected backorder (EBO) model invokes Palm's Theorem in every resupply situation including depot resupply to a base.

Also, it is assumed that the system is in steady state, i.e., the demand process is stationary.

Model Objective. As in MOD-METRIC, the minimization of worldwide expected base backorders subject to a budget constraint is the objective of this model. At the same time we wish to maximize aircraft availability. This model provides a more general case because it can handle multiple LRUs and SRUs. It also models common components and includes a repair option. With its multi-indenture feature it can model up to five levels of indenture.

Problem Solution. The AAM method for calculating availability makes use of the measure of expected backorders. For this model, availability rate is the percentage of aircraft with a complete set of reparable parts. The definition of availability, as given before, does not consider on-aircraft maintenance, scheduled or unscheduled, and shortages of consumables. For the purpose of the model, an available aircraft is one with no LRU backorders outstanding.

The AAM computes the availability rate resulting from a given inventory of spares in a two-step process. First, it computes the expected backorders for each component on the aircraft. Second, it computes the probability of one or more of those expected backorders occurring on an aircraft.

An ordered list of components is produced by the computation procedure of the model. Buying from this list, in the order indicated, will yield the desired availability rate for the aircraft type at minimum cost.

A marginal analysis technique is the optimization procedure used by the model. Candidate units for procurement are ranked in terms of decreasing benefit per unit cost, where the benefit is defined in terms of the increase in availability rate which would occur if that spare unit were added to the inventory.

Proper consideration of commonality is crucial if results are to be meaningful, simply because there is so much of it, and the AAM model is accounting for it. The construction of the availability curves proceeds as before, with one important difference: the curves are no longer independent. The contribution from a spare unit of a common component appears in several curves, reflecting the availability improvement that the unit's procurement would bring to several aircraft types.

The AAM may be run with a repair option, which gives it the capability to trade off depot repair with procurement of reparable items.

Components are classified by level of indenture. Not all them are on the first level of indenture, i.e., that they are applied directly to the aircraft. All components on the lowest level of indenture are processed first. The next higher level of indenture is then processed, considering the effect of the lower indenture level. In effect, each component on the higher level has a curve of cost versus lower level support, where the lower support is measured in terms of the number of higher level components awaiting lower level subassemblies. Investment in spare units of the higher level component is traded off against an equal investment in its subassemblies to minimize expected backorders of the higher level component. The process is then repeated at the next higher indenture level, and so on.

Assumptions

The overview of the METRIC, MOD-METRIC, and AAM models gives the frame work required to state our assumptions for the inventory model to be proposed for the Portuguese Air Force.

Two major groups of assumptions are considered. One concerning the demand and resupply process. Another grouping together the assumptions for the repair process.

The total spares inventory needed to support a flying hour program varies linearly with the program; that is, item failures are linearly related with flying hours.

The demand for recoverable items observed at air bases has high variability and usually produces variances that exceed the means. Furthermore maintenance personnel sometimes order several units at once, i.e., they do batching orders. Thus, it seems appropriate to assume that demand is well described by a compound Poisson process.

For computational advantages we follow Sherbrooke (21), that assumes demand at each base has the same variance to mean ratio, though different means. By assuming this we obtain a compound logarithmic Poisson process at depot with that variance to mean ratio.

Resupply time is assumed to be independent of demand. Demand is considered to be stationary; that is, arrival rates and variance to mean ratio are assumed to be constant over a long period of time.

Concluding the set of assumptions for the demand and resupply processes we consider that no lateral resupply is available. Base orders only can be filled by an item supplied by the depot.

Regarding the assumptions for the repair process, we have that component failures are independent of each other. The repair process is independent of maintenance workload. By stating this we assume that all support equipment and maintenance manpower needed to repair the failed components will be available. If workload increases, more manpower and equipment is added to the repair process. Repair times are statistically independent. One component repair time does not depends on another component repair time. And lastly we assume that no batching occurs in the repair process.

Description of the context in which the model will be used.

The aircraft inventory of the Portuguese Air Force is spread into a relatively large number of aircraft types which have a small number of aircraft in each type. Aircraft subtypes are not significant. As a result of this situation, in some cases all aircraft from one type are deployed to only one air base. Thus, only one aircraft type operating at one base will be modeled. Peacetime operational activity is considered for this aircraft type. This implies that flying activity is steady over a long period of time. No commonality of components exists with other aircraft types. This means that lateral resupply is ignored in the model.

The Portuguese Air Force supply system has one central depot, one industrial repair facility with repair capacity for almost all aircraft recoverables, and several operational bases geographically located not far from the depot. Spare parts carried in the inventory are classified as consumables or recoverables. For each item in stock a decision is made in what concerns the supply stock level of the item at each base. In general, when an item has high demand, managers assign to it a requisitioning objective level. Items with a level assigned are controlled under the (s,S) inventory policy. Almost all consumables fall into this category. When an item has a level established, any time that the quantity on hand plus on order minus backorders is less than or equal to the reorder point, a order is placed to bring the stock to the requisitioning objective level. For all other spare parts each time that the stock is decreased by one a order is placed to replenish the stock level. This management technique is the well known (s-1,s) inventory policy.

The repair cycle is based on a two echelon system. Air bases have limited repair capability. Usually this repair capability is limited to replacement of LRUs and repair of a few SRUs. When an item fails and it is declared NRTS, the item is sent to the depot for repair and an order is placed for a replacement. When the item is received by the depot it is sent either to the repair facility, if it has the required repair capability, or to one of the contractors. As the repair finishes it is returned to the depot.

The management of the supply system is the responsibility of the Supply Directorate under the Logistics Command of the Air Force. The budget assigned to spare parts is divided by several departments and each one manages one class of components for all aircraft types. Mechanical components, electronic components and aircraft armament and weapons have independent budget categories.

The hierarchical relationship between spare parts carried in the inventory is defined by two levels of indenture. The first level includes all components that are directly applied to the aircraft; that is, all LRUs. The second level includes all other components. SRUs and their subcomponents are included in this level. In general, no commonality of recoverables exists among different aircraft types. This means that lateral resupply can not be incorporated in the model.

Economic restrictions and political difficulties have had serious implications on the approval of the budget for the public services and defense. For the last several years important cuts had been inflicted on the proposed budget. Besides these cuts, the budget has been approved late in the year. While the new budget awaits approval, expenses can only be incurred up to the amount that corresponds to a month in the previous budget. As a result of this situation, managers can not process all required spare parts acquisitions to replenish stocks. They hold more money for priority acquisitions that can arise in the meanwhile. In the short run, no big changes are detected in the support of the operational activities because of the slow reaction of the supply system. The reaction shows up in the long run when stocks start to be short. In this condition consumables tend to be managed as (s-1,s) components. Managers just buy the quantities required to satisfy orders placed by maintenance.

Summary

In summary we can say that the analysis of the METRIC, MOD-METRIC, and AAM models gave the necessary background to formulate the assumptions for the model to be suggested for the Portuguese Air Force. Together with the scenario described we can state that the proposed model is a two-echelon, two-indenture model with multiple items and subject to a budget constraint. The next chapter is devoted to the development of this model.

III. Methodology

Until this point all the research effort was directed to the collection and review of major theories and models that are used in inventory systems and, in one way or another, related with spare parts calculation and aircraft availability. In this chapter the two-echelon, two-indenture model for one aircraft type, proposed for the Portuguese Air Force, will be developed. Collection of new data, run procedures for the model and other requirements will be discussed in the implementation section. This chapter is concluded with the analysis of the proposed research objectives and reviewing how the model answers to the research questions.

Model Development

As a result of the discussion of the METRIC, MOD-METRIC and AAM models, it is evident that availability rates and backorders per aircraft are closely related. Methods of computing expected backorders (EBOs) for a component in a multi-echelon, multi-indenture system, as used by METRIC and MOD-METRIC, are well known. This technique will be used once again. However, to ensure adequate support of end items, it is necessary to go beyond consideration of component oriented measures and to calculate the probable effect of these component shortages upon aircraft. The AAM accounts for this and its orientation will be followed. "The conceptual framework and mathematics" discussed by O'Malley (11) will be the guidelines on the development of the model. Thus, the availability rate resulting from a given inventory of spares is computed in a two-step process. First, expected backorders for each component on the aircraft will be computed. Second, the probability of one or more of those expected backorders occurring on an aircraft will be evaluated. The next step in the development of the model is to address the optimization question: given a budget constraint, what spares should be procured to attain the best possible availability rate? The marginal analysis technique is the optimization procedure used and it will be discussed. The last step and the ultimate goal for the model is the discussion of the algorithm used to generate the aircraft availability curve and the shop lists.

Expected backorder. The first step is to compute the number of expected backorders for each component and for the projected level of spares for the component if no further procurements are made. The computation of component expected backorders (EBOs) is derived from Sherbrooke's METRIC: Multi-Echelon Technique for Recoverable Item Control (20).

The model computes, one at a time, component EBOs using the following input (for each component):

- The expected number of units in base repair (the base repair pipeline)
- The expected number of units in depot repair (the depot repair pipeline)
- The expected number of units in transit from the depot to a base (the order and ship pipeline)
- The number of bases at which demands for the component occur (number of users).

The model produces an array of the component EBOs as a function of the spares level.

The pipeline used in the EBO model is computed by multiplying the appropriate demand rate by the corresponding resupply time. For example, the base repair pipeline equals the base repair delay demand rate (DDR) times the base repair time in days (base repair pipeline = Base DDR * BRT). The Base DDR equals the total DDR times (1 - NRTS) (i.e., the percentage of repairs which are beyond the capability of the base repair shop). Similarly, the depot repair pipeline = Total DDR * NRTS * DRT, and the order and ship pipeline = total DDR * NRTS * OST, where DRT is the depot repair time and OST is the order and ship time. Palm's Theorem then implies that the distribution of the number of units in resupply in any particular resupply segment is then given by a Poisson distribution whose mean is the corresponding pipeline.

The EBO model can allow for uncertainty in the mean demand rate when the demand process is Poisson but the mean demand rate is not exactly known. This is appropriate for the model since component demand rates can change over time. The EBO model uses a Gamma distribution to describe the probability distribution of the mean of the demand rate. Combining this with Palm's Theorem gives a negative binomial distribution for the number in any particular resupply pipeline. With complete certainty of demand rate, the Gamma reduces to a point distribution, and the negative binomial reduces to the Poisson (11). The formula for expected backorders for a particular component at a particular site is:

$$EBO = \sum_{x>s} (x-s)p(x) \qquad (1)$$

where,

s is stock level

x is the number of units in resupply for that site (including in repair at the site, on order from another site, and all other forms of due in to that site)

p(x) is the probability of having x units in resupply (a Poisson or negative binomial probability distribution).

The component EBO produced by the model is the total of component's EBOs at all bases. The depot EBO is coupled to this total by its impact on the number in resupply at the bases.

For a given component, the model computes the total EBO for many different total asset levels. For each asset level, the model considers every possible way to distribute those assets between base and depot and selects the distribution with the lowest total EBO. The computation solution consists of five stages:

1. Compute for each base the average order and ship time.

2. For each level of depot stock and each base, compute the EBO reduction as function of base stock.

3. For each level of depot stock determine the optimal allocation of the first, second, ... unit of stock to the several bases so as to minimize the sum of EBO at all bases. This is accomplished by marginal allocation.

4. EBO and total stock s is collapsed into one dimension. Select minimum EBO for system.

5. Using marginal analysis the next investment is allocated to the item which produces the maximum decrease in EBO divided by unit cost.

The model first computes the EBO for each spares level at the depot using Eq (1). The distribution used is a negative binomial whose mean is the depot repair pipeline.

The computation of EBOs at a base is similar to the computation of depot EBOs, and depends on the calculated depot EBO. From a base's perspective, the depot EBO is a resupply pipeline. The model views a backorder at the depot as a unit in resupply to a base in the "depot delay pipeline." The total base resupply pipeline equals the base repair pipeline plus the order and ship pipeline plus the depot delay pipeline (Total base resupply pipeline = base repair pipeline + order and ship pipeline + depot EBO). The mean of the distribution of the number in resupply at a particular base equals the total base resupply pipeline divided by the number of bases. The EBO model makes an important simplifying assumption here: all bases have equal average demand rates. The model then need only consider allocations of assets where each base gets the same number of spares (i.e., if there are three bases, the total number of spares allocated to the bases can only be 0,3,6,9.... As each of the first 3 spares gives the same EBO reduction as it is placed at each base, the EBO for 1 or 2 spares at the bases can be obtained by a linear interpolation between the EBO total for 0 spares at base level and the EBO total for 3 spares at base level). The total EBO equals the EBO at one base, as computed by Eq (1), times the number of bases. Thus, a given spares level at the depot determines the base resupply pipeline and the resulting total EBO for all spares levels at the bases. The optimum distribution of spares between bases and depot is determined by comparison. As the resulting EBO for a given distribution of spares is calculated, it is compared with other EBOs for the same total (base and depot) spares level. The optimum distribution is that which yields the least EBO for the given level.

Notation. The following notation will be used in this chapter:

A — Total aircraft availability.

AC — Total number of aircraft.

C(i) - Procurement cost of component i.

I(i,s) - Improvement factor of component i at spares level s.

N - Number of different components installed on the aircraft type.

n(i) - Spares level for component i; or s = n(i).

QPA(i) - Quantity per application of component i on the aircraft type.

q(i,s) - availability contribution of component i, at spares level s.

S(i,s) - Sort value of component i at spares level s.

T(i) - Total number of components i installed on the aircraft type.

<u>Availability rate</u>. Consider N as the total number of different components installed on one aircraft type, and T(i) as the total number of units of component i installed on the aircraft type, or the total number of "slots" on the aircraft type which should contain a functioning unit of the component. A backorder for the component i results in an empty slot, a "hole" on the aircraft. With a given spares level n for each component i, s=n(i), the probability that any slot is backordered is EBO(i,s)/T(i), assuming that backorders are uniformly distributed among slots. The probability that a slot is not waiting for a spare is 1-EBO(i,s)/T(i), and, for an aircraft with a quantity per application (QPA) equal to a, the probability that the

aircraft is not waiting for spare i is

$$q(i,s) = (1 - EBO(i,s)/T(i))^{a}$$

This probability is called the component aircraft availability.

Assuming independence of backorders between components, the probability that a random aircraft is not missing any of its components is the product of all the individual component probabilities

$$A = \prod_{i=1}^{N} q(i,s) \qquad (2)$$

Since an available aircraft is defined to be one which is not missing any component, A is also the probability that the aircraft is available, the availability rate for the aircraft type. Since the definition of aircraft availability involves expected backorders, it is not surprising that there is a fairly simple relationship between projected availability, A, and expected backorders per aircraft (EBO/AC). In fact, for each aircraft type, it is approximately true that $EBO/AC = -\ln A$ where EBO is the sum of all LRU expected backorders for the aircraft type. Note that the relationship does not depend on fleet size or aircraft complexity.

The mathematical justification for this relationship is given by O'Malley (11). Let EBO(i,s) be the number of expected backorders for component i at spares level s, let QPA(i) be its quantity per application on the aircraft type, and let AC be the number of aircraft. Thus, the availability rate is given by

$$A = \prod_{i=1}^{N} \left(1 - \frac{EBO(i,s)}{AC * QPA(i)}\right)^{QPA(i)}$$

where the product is taken over all first indenture level items (LRUs) on the aircraft.

The power series for exp(-EBO(i,s)/(AC*QPA(i))) is

$$1 - \frac{EBO(i,s)}{AC*QPA(i)} + \frac{1}{2!}\left(\frac{EBO(i,s)}{AC*QPA(i)}\right)^2 - \frac{1}{3!}\left(\frac{EBO(i,s)}{AC*QPA(i)}\right)^3 + \ldots.$$

Since

$$\frac{EBO(i,s)}{AC*QPA(i)}$$

is typically small, we may ignore the higher order term and write

$$\exp(-EBO(i,s)/(AC*QPA(i))) = 1 - \frac{EBO(i,s)}{AC*QPA(i)}$$

Then

$$A = \prod_{i=1}^{N} \left(1 - \frac{EBO(i,s)}{AC*QPA(i)}\right)^{QPA(i)}$$

$$= \prod_{i=1}^{N} [\exp(-EBO(i,s)/(AC*QPA(i)))]^{QPA(i)}$$

$$= \prod_{i=1}^{N} \exp(-EBO(i,s)/AC)$$

$$= \exp\left(\sum_{i=1}^{N} -EBO(i,s)/AC\right)$$

$$= \exp(-EBO/AC)$$

So we have A = exp(-EBO/AC) or, equivalently, EBO/AC = -ln A.

The Optimization Procedure. Marginal analysis technique is the optimization procedure used in the model. Candidate units for procurement are ranked in terms of decreasing benefit per unit cost, where the benefit is defined in terms of the increase in availability rate which would occur if that spare unit were added to the inventory.

Starting at the level of spares s, the procurement of an additional spare unit reduces the expected backorders from EBO(i,s) to EBO(i,s+1). It increases the probability that an aircraft is not missing a unit of this component from q(i,s) to q(i,s+1).

The availability rate of the aircraft type, before procurement of the first additional unit, is

$$A = \prod_{i=1}^{N} q(i,s)$$

$$= [\prod_{i=1}^{N} q(i,s)] * q(j,s), \qquad \text{for } i \neq j$$

The availability rate after the first additional unit of component j (spare unit s+1) is procured is

$$A' = [\prod_{i=1}^{N} q(i,s)] * q(j,s+1), \qquad \text{for } i \neq j$$

Thus, the ratio of the new to the old availability rates,

$$A'/A = q(j,s+1)/q(j,s)$$

depends only on the spares level of component j.

We call this ratio I(j,s+1), the improvement factor due to unit s+1 of component j. In general,

$$I(i,s) = q(i,s)/q(i,s-1)$$

If C(i) is the procurement cost of component i, define the sort value of the nth unit of component i, S(i,s), to be

$$S(i,s) = \ln(I(i,s))/C(i)$$
$$= \ln(q(i,s)/q(i,s-1))/C(i)$$

As might be expected from the nomenclature, the sort value is the measure of benefit per cost that is used to sort the candidate units for procurement. For the marginal analysis a ordered list of sort values in descending order must be generated. This list is formed based on the diminishing differences in the component improvement availability.

Thus, we convert the problem to the maximization of the sum by taking the natural logarithms of the product form of the availability, as expressed in product formula Eq (2), and maximizing

$$\sum_{i=1}^{N} \ln q(i,s)$$

for the cost C(i).

Calculations to this point comprise an array for each component i and for all spares levels s, summarized in Table I.

Number Spares	Expected Backorders	Component Aircraft Availability	Improvement Factor	Sort Value
s	EBO(i,s)	q(i,s)	-	-
s+1	EBO(i,s+1)	q(i,s+1)	I(i,s+1)	S(i,s+1)
s+2	EBO(i,s+2)	q(i,s+2)	I(i,s+2)	S(i,s+2)
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

TABLE I. Component Information

The "starting availability rate," As, for the aircraft type is calculated by Eq (2). Thus

$$As = \prod_{i=1}^{N} q(i,s)$$

where the index i includes all N different components applied to the aircraft type where s is the projected spares level for component i if no further procurement are made.

The first unit on the shopping list will be that with the highest sort value, say unit s+1 of component j. Since $S(j,s+1) = \ln(q(j,s+1)/q(j,s))/C(j)$, the availability rate after this unit is procured is given by

$$
\begin{aligned}
A &= As * \exp(C(j)*S(j,s+1)) \\
&= [\prod_{i=1}^{N} q(i,s)] * \exp(\ln(q(j,s+1)/q(j,s))) \\
&= [\prod_{i=1}^{N} q(i,s)] * (q(j,s+1)/q(j,s)) \\
&= [\prod_{i=1}^{N} q(i,s)] * q(j,s) * (q(j,s+1)/q(j,s)), \text{ for } i \neq j \\
&= [\prod_{i=1}^{N} q(i,s)] * q(j,s+1), \quad \text{for } i \neq j
\end{aligned}
$$

This is the product of the item availability reflecting the new spares levels. We now add the next item, the one with the second highest sort value to the shopping list. The general form of the above relation is then used to calculate the availability rate after this unit and each subsequent unit are added to the list.

$$Anew = Aold * \exp(S(j,s)*C(j)) \tag{3}$$

when the nth unit of item j is added to the list. Continuing in this way, we obtain a shopping list of which Table II is a hypothetical example.

STARTING POSITION: .6666 AVAILABILITY 5 UNITS OF A AND 1 OF C ON HAND			
No. of Units Component	Unit Cost	Cumulative Cost	Availability Rate
6 th A	$1,598	$1,598	.6667
1 st B	2,300	3,898	.6669
2 nd C	10,400	14,298	.6674
2 nd B	2,300	16,598	.6676
1 st D	13,800	30,398	.6678
7 th A	1,598	31,996	.6679
.	.	.	.
.	.	.	.
.	.	.	.

TABLE II. Shopping List

An availability rate of 66.78 percent can be attained by buying the first five entries in Table II (1 unit each of components A, C, and D, and 2 units of component B). This availability rate is attained at a cost of $30,398.

<u>Shopping List and Cost Versus Availability Curves - The Algorithm</u>. In the first section we examined the theory behind the development of aircraft availability curves and the shopping list. Then we discussed the usefulness of the aircraft availability curve. Now, we will discuss the mechanics of constructing an aircraft availability curve and a shopping list.

The first point on the aircraft availability curve is determined by buying enough spare units to fill the pipeline or a specified percentage of the pipeline or buying additive quantities before beginning marginal analysis. Sometimes, projected asset levels are negative, due to condemnations. In such cases, enough spares must be procured to reach a zero level. These procurements are made automatically and the cost accumulated into a "sunk" cost.

The contribution of each component to the aircraft availability, q(i,s), is computed for each spares level s, as is the sort value, S(i,s), for the nth spare unit. The starting component availability, q(i,s) contributes to a computation of the starting availability rate for the aircraft type. When processing for each component is complete, component summary data are written into a file and saved for later processing. These data contain a header identifying the component, its sunk cost, starting EBO and availability, unit cost, and records for each additional spare consisting of sort value S(i,s) and cost C(i). For each additional spare unit of the component, a record identifying the component, its cost, and the sort value of that spare unit is written into a sort value file and saved.

When processing for all components is complete, the sort value file is then sorted. After sorting, all records fall in order of descending sort value.

The (sorted) sort value file is now processed to construct the availability curves. The sunk cost and starting availability form the first point on the curve. As each record is read from the sort value file, the cost is accumulated, and the availability resulting from that procurement is calculated according to Eq (3).

To obtain the shopping list corresponding to a point on the curve, the data in the component summary data file is used. Given a cost or availability rate to identify a point on the curve, the curve file contains a sort value, S, associated with that point. Every spare unit which had a sort value greater than S must be procured to attain that availability rate. These can be determined from the data on the component summary data file. Adding the number of units of the component bought as part of the sunk cost yields the total buy quantity for the component.

Since the model considers only two levels of hierarchical relationship, all components are classified as LRUs or SRUs. The computation of the aircraft availability and the shopping list procedures are repeated for each level. SRUs are processed first. LRUs are then processed, considering the effect of the SRUs.

As a result of this computation, an expected backorder array is obtained for each LRU. This array is similar to that in table I with one difference -- some entries of the array correspond, not to buying an additional spare unit of the LRU, but to investing an amount of money equal to the cost of the LRU into the SRUs. The curve of cost versus availability rate is constructed as for LRUs. q(i,n) is defined as before, n is now the number of equivalent SRUs (buying from SRU shopping list until the cumulative cost equals n times the LRU unit cost) of component i, the component itself or its SRUs. The product formula for aircraft availability is as before, Eq (2), except that only the terms for LRUs are used. The effect of the SRUs is already included in the expected backorder total for the LRUs and their corresponding LRUs contribution to availability rate.

Finally, the two shopping lists are merged and split by budget code. There are a total of ten different budget codes. Cumulative quantities for total asset position, total number of components bought by the model, and total expenditures are identified per budget code. For each aircraft type, procuring the total number of spare parts listed across all budget codes, the Portuguese Air Force will achieve its availability goals at minimum cost.

<u>Structure of the model</u>. Two batch files, AAM and SHOP, are the backbone of this model. These files are executed in sequence. The run flow of the model, in what concerns the main programs executed, is depicted in Figure 3, for the AAM file, and Figure 4, for the SHOP file. The first file, appendix A, starts with the run of the program that sets the initial conditions for the variables. Following the set up of the values, the main program for subassemblies performs the analysis of the SRU components. In the next step, the SRU summary data file is sorted. The sorted file is used as

input for the following run of the main program. At this time, all LRUs are subjected to the analysis of the main program. For this analysis the contribution of the SRUs is considered. Once again, the sort program is run for the LRU summary data file. This batch file terminates its execution with the run of the program that generates availability curves.

```
 -------------------------------------------
| Initialization of run variables (HIGHSET) |
 -------------------------------------------
                     |
 -------------------------------------------
|   Analysis of Level 2 Components (SAM)    |
 -------------------------------------------
                     |
 -------------------------------------------
|    Sort Results of Level 2 (SPLITSRT)     |
 -------------------------------------------
                     |
 -------------------------------------------
|   Analysis of Level 1 Components (SAM)    |
 -------------------------------------------
                     |
 -------------------------------------------
|    Sort Results of Level 1 (SPLITSRT)     |
 -------------------------------------------
                     |
 -------------------------------------------
|   Generate Availability Curve (CURVE)     |
 -------------------------------------------
```

Figure 3. Main Batch File (AAM)

The second batch file, appendix B, runs the shopping list programs, one for each level of indenture. When these lists are complete the last program is executed and the shopping lists by budget code are produced.

```
 ---------------------------------------------
| Shoplist for Level 1 Components (SHOP1)     |
 ---------------------------------------------
                      |
 ---------------------------------------------
| Shoplist for Level 2 Components (SHOP2)     |
 ---------------------------------------------
                      |
 ---------------------------------------------
|      Shoplist by Budget Code  (SHOPB)       |
 ---------------------------------------------
```

Figure 4. Batch file for Shoplists (SHOP)

Implementation of the Model

This model requires several data inputs. Two types of inputs are considered. In the first, the model interacts with the user in a system characteristics question/answer sequence. The required inputs are:

- The Budget Year. This must match the year of the component data bases. May be any number from 80 to 99.
- The Variance-to-Mean Ratio. The component demand process may be modeled as Poisson or negative binomial. A variance-to-mean ratio of 1.0 denotes a Poisson demand process. A higher ratio denotes a negative binomial with that variance-to-mean ratio.
- The Pipeline Buy Percentage. The model allows for different alternatives in filling the pipeline.
- The System Name. This is the name for the aircraft type under analysis.

- The Flying Hour Program. The annual flying hour goal established for the aircraft type must be entered at this time.
- The Number of Aircraft Deployed. This is the total number of aircraft that the model is to compute logistics support for.
- The Number of Bases. The supply system consists of one central repair and resupply depot and a number of retail repair and resupply sites, called Bases or Intermediate Maintenance Activities.

In the second set all component characteristics are inputed using data files. The component data should be stored in two files. One file for each level of indenture. The files contain one record for each component. The fields in each record are:

- The Component Name. Usually, the National Stock Number (NSN) is used.
- The Component Cost. The unit procurement cost.
- The Base Repair Pipeline. The total expected number of units in repair at all bases. This may be computed by multiplying the average number of units inducted into repair in all bases per day by the average repair time in days.
- The Order and Ship Pipeline. The expected number of units in transit from the depot to the bases. This

may be computed by multiplying the average number of units ordered from the depot by the bases per day times the average order and ship time in days.

- The Depot Repair Pipeline (including the Retrograde Ship Pipeline). The expected number of units in depot repair (or sent to a contractor) or in transit from bases to the depot. This may be computed by multiplying the average number of units sent to the depot for repair per day by the sum of the depot repair cycle time in days, including the retrograde ship time.
- The Condemnation Replacement Pipeline. The expected number of units on order from the manufacturer. This pipeline may be computed by multiplying the average number of orders per month by the procurement leadtime (including administrative leadtime) in months.
- The Total Asset Position. This is defined as the number on hand plus the number due in (on order), minus the number due out (backorders).
- The Name of the Next Higher Assembly. For level 1 components, this should be the aircraft type name. For a level 2 component, this should be the name of the level 1 component it is installed on.
- The Fixed Level. If the asset position is to be raised to a specific level as a sacrosanct policy

decision specific to this component, this level may be input here. The model will buy up to this level and record the cost accordingly before beginning the optimization algorithm. The algorithm cannot reduce the level below this point.

- Budget Code. Though the model optimizes total budget, it produces budget subtotals by code. This allows management to be done by department while a system management concept is followed.

One more input data file is required by this model. The average flying hours flown by year, which is the baseline for the computation of the components pipelines, is the value to be stored in this file. An example of these files is presented in appendix C.

While part of these component data exists in the actual data bases of the Portuguese Air Force, several changes are needed in order to implement this model.

Meeting the Research Objectives

With the development of this easy-to-use computer model, which runs quickly and requires a minimum amount of input data, the research objectives were met. The relationship between aircraft availability and the optimal level of spare parts to carry on stock for this aircraft type, was found. With this model as a tool, managers can determine the effect of proposed changes in the spare parts budget on aircraft availability. Also, they can submit a budget proposal based on the flying hour program defined for each aircraft type.

Summary

This chapter was concerned with the development of the model and its implementation as well as how the model meets the research objectives. It is the intent of the following chapter to do the analysis of the results of the model and to perform the model verification and validation.

IV MODEL AND RESULTS

In Chapter III the model was described in a general overview. This chapter presents more detail in order to understand how to use the model and begins with a narrative description of the model. Following this description a discussion on verification and validation of the model will be presented. A baseline scenario is defined and the model tested against the historic data given by the Directorate of Supply of the Portuguese Air Force. This chapter concludes with discussion of results and analysis of sensitivity of the model to different levels of the flying hour program, number of aircraft and availability.

Narrative Description

The model consists of two main parts, the AAM batch file, appendix A, which generates the component summary data files and availability versus cost curve, and the SHOP batch, appendix B, file that does the shopping list as described in Chapter III. The AAM file runs four programs. All of the programs are coded in Fortran. They are presented in appendix D. HIGHSET is the first one. This program does the inputs for the run. Parameters for the run and weapon system characteristics are the inputs required at this step. The directory where the data files exists, the budget year, the variance-to-mean ratio, the sacrosanct pipeline buy percentage, the weapon system name, the flying hour program, the number of aircraft deployed and the number of bases are such inputs. Calling UPDATA, this program adjusts the pipelines according to the flying hour program input. After gathering these inputs, the model processes the level 2 components. SAM, the main program for subassemblies, does all of the computations:

- pipelines are filled, it buys the starting asset position after sacrosanct buys.(main program)
- EBO computations (subroutine LUMPCMP)
- Marginal analysis (subroutine MARG)
- Computes SV=(EBO(.)-EBO(.+1))/cost, the sort value for all components.

Next, SPLITSRT takes the execution. It sorts all component sort values. The first record in the file will be that with the highest sort value, the second item is the one with the second highest sort value and so on.

After level 2 components are processed, the execution returns to SAM for processing of level 1 components. Next, the results are sorted by SPLITSRT and the cycle for the analysis of components is finished.

Before AAM, the first batch file, is terminated, CURVES, the next program in the sequence, generates a table of availability/cost values which constitute a curve of availability versus cost. The user may then choose an availability/cost point from anywhere on that curve -- not just from the points printed -- for which the cost subtotals by budget code will be printed and for which a shopping list may be run. If an availability point is selected, the model gives the cost associated with this decision. If the level of investment is the goal to be achieved, the program gives the projected availability for the selected option. This curve may be saved for later analysis. When AAM terminates its execution the user may then generate the shopping list corresponding to the decision by running SHOPLIST which initiates the second batch file, or may rerun the CURVES program by typing "curves". CURVES can be rerun as many times as the user wants until the final decision is accepted.

When the first program finishes execution, if the user types SHOPLIST, the second batch file is initiated. The shopping lists are generated by analyzing level 1 components followed by the analysis of level 2 components. The shopping list program for level 1 components, SHOP1, generates the level 1 shopping list from analysis of the component summary data file (called RESULTS which is generated by SAM). When finished, SHOP2, a shopping list program identical to SHOP1, generates the level 2 component data file. The model finishes its execution with SHOPB, a program which generates the output data files using shopping lists from both level 1 and level 2 components.

The user may generate a number of different shopping lists, each corresponding to a different decision, by rerunning CURVES and SHOPLIST as many times as desired. Each time SHOPLIST is run, it generates the shopping list for the most recent decision. However, each new shopping list erases the previous one. So if the user wants to save multiple shopping lists, they must be copied onto backup files.

Verification and Validation of the Model

Verification is the process of determining the model works as intended while validation is the process of determining the model accurately portrays the real system being modeled (22:10). The purpose of model verification is to assure that the conceptual model is reflected with accuracy in the computer code.

Before the description of the procedures used in model verification and validation, it is important to refer to the results of tests conducted by Logistics Management Institute and the Air Force Logistics Command (AFLC) in order to validate the AAM model in use by AFLC. The model was tested in 1977-1978. Several extensive and carefully monitored tests of the AAM model were conducted, and they demonstrated that the model does provide a valid way of relating an inventory of spare parts to the availability rates of the aircraft types which those spares support. Also, it could accurately forecast aircraft availability rates.

The conclusion reached by AFLC was that "... the model is a valid means of computing the probability that an aircraft will not be missing a recoverable part" and " ... the LMI model can be used to determine a reasonable indication of actual aircraft availability "(18). The validity of the AAM model is important . The importance of this conclusion is given by the close relationship between this model and our model. Recall that our model is a simplification of the AAM model, which uses the techniques discussed for METRIC and MOD-METRIC. As a verification procedure the code was checked against the original model. The AAM model code as well as its microcomputer version, were the support to the development of the code for this model. The code was carefully checked and documented to insure against any mistake. The model was checked to see if it executes as the modeler intended by doing some manual checking calculations.

As a first step in running the model, sample data files were created because no other input data was available at that time. With this data, some sensitivity analysis was performed to check the model. For example, increasing the pipelines of the components while keeping the availability rate at the same level, the shopping list is increased. Several debugging statements were included in the code. These statements were used to help in verification of the model flow.

Validation of the model is a more difficult task. Ideally, a model can be validated by using historic inputs and then comparing the model outputs to the historic outputs. For this purpose, historic data was requested from the Portuguese Air Force Logistics Command. The request for this data caused several problems for the managers of the aircraft type in the Directorate of Supply. The data requested was not easily available. In spite of the initial difficulties, two data files were received. They are the component data files for the CHIPMUNK aircraft type. As a starting point in the validation process, our efforts were aimed at considering the reasonableness of the model outputs for the given inputs. Initial runs were executed to test for this reasonableness. The observed output values were determined to be near the expected values for such measures as total expenditures given a level of availability. In addition, changes in the output values occurred in ways expected as the inputs were varied. For example, for a given level of availability, total costs increased when pipelines are increased.

<u>Selected aircraft type</u>. Among the aircraft inventory in the Portuguese Air Force the CHIPMUNK is the aircraft type selected. A total of 36 aircraft are in the inventory. The CHIPMUNK is a small trainer with a small number of components, and its data files are easy to handle. Component data are obtained from the Supply and Maintenance Integrated Management System, such as name, unit cost, condemnation and replacement pipeline, next higher assembly, and budget code.

The base repair pipeline and depot repair pipeline were computed using the maintenance repair reports from base and depot, respectively. For these computations an average base repair time of 10 days and an average depot repair time of 120 days were considered for each reparable. Total assets were calculated as the total on hand at base plus the total on hand at depot, including all components under repair, minus backorders.

Under the Directorate of Supply of the Portuguese Air Force Logistics Command the management of the supply system is organized by categories of classes of material such as aircraft, electronics, armament and others. A budget code is assigned to each of these working areas. Budget codes 1 to 4 were given to level 1 components managed by areas 1 to 4. For level 2 components budget codes 5 to 8 are related with management areas 1 to 4, respectively. Budget codes 9 and 10 are not used. For this aircraft type budget codes 2,6 and 7 were not assigned.

Included in the package received from the Portuguese Air Force there was information related with the availability rates, number of aircraft and flying hour program for the CHIPMUNK aircraft type. From the relevant information concerned with the fiscal year 1986, the following data elements were computed:

- 30 aircraft deployed to one base. This is the only base operating this aircraft type.

- 2400 total flying hours were flown during this period.
- average aircraft availability computed to 70%.

These values are defined as the baseline scenario for the validation process of the model. Thus, 30 aircraft operated by one base, with an average of 70% of availability for a flying activity of 2400 hours per year is the baseline scenario.

The validation goal is to compare model results with the level of budget assigned to this aircraft type. For the fiscal year of 1986 a 20,000,000$00 (escudos, the Portuguese currency) were spent in procurement of spares for the CHIPMUNK aircraft type. As assumed before, this total expenditures figure is the minimum investment required to replenish stocks. During recent years, due to a strong budget constraint, the investment in stocks has been restrictive and no increase in stock levels has been accomplished. As a result, the flying activity is kept at low levels and was constant over these last years.

Analysis of the input data. Data was received on a floppy disk. All component data records are grouped in two files. DATA.yy1 is the file name for level 1 components for the yy year. DATA.yy2 is the equivalent file for level 2 components. With the data files received by floppy disk it was easy to load the contents on the computer system. The following problems were found in the CHIPMUNK data files:

- data on pipelines for level 2 components was missing. No information was found in order to fill this field.
- some components had multientries for the same application. As a result of this, the model only boughts the first of these items on the list.
- components with application on itself. The model does not model this situation. These records are ignored.

<u>Run for baseline scenario</u>. The analysis of the data suggested some changes to the contents of the files. For example, in the file for level 2 components the quantities on acquisition for replacement, under the condemnation and replacement pipeline, were summed with the total assets since this total was not reflecting the spares on order. In the level 1 data file, total assets included the aircraft inventory. 36 additional units are in the inventory while they are installed on the aircraft. They are not available as spares. For this model, only spare parts are counted. Thus, this field was adjusted. With these corrections to the data files the model was run for the baseline scenario of 2400 flying hour program, 30 aircraft and 70% availability. Figure 5 presents a plot of the availability versus total expenditures curve generated by the model. An additional expenditure of 11,041,551$00 is the recommendation of the model. The results are presented in table III.

AVAILABILITY vs EXPENDITURES

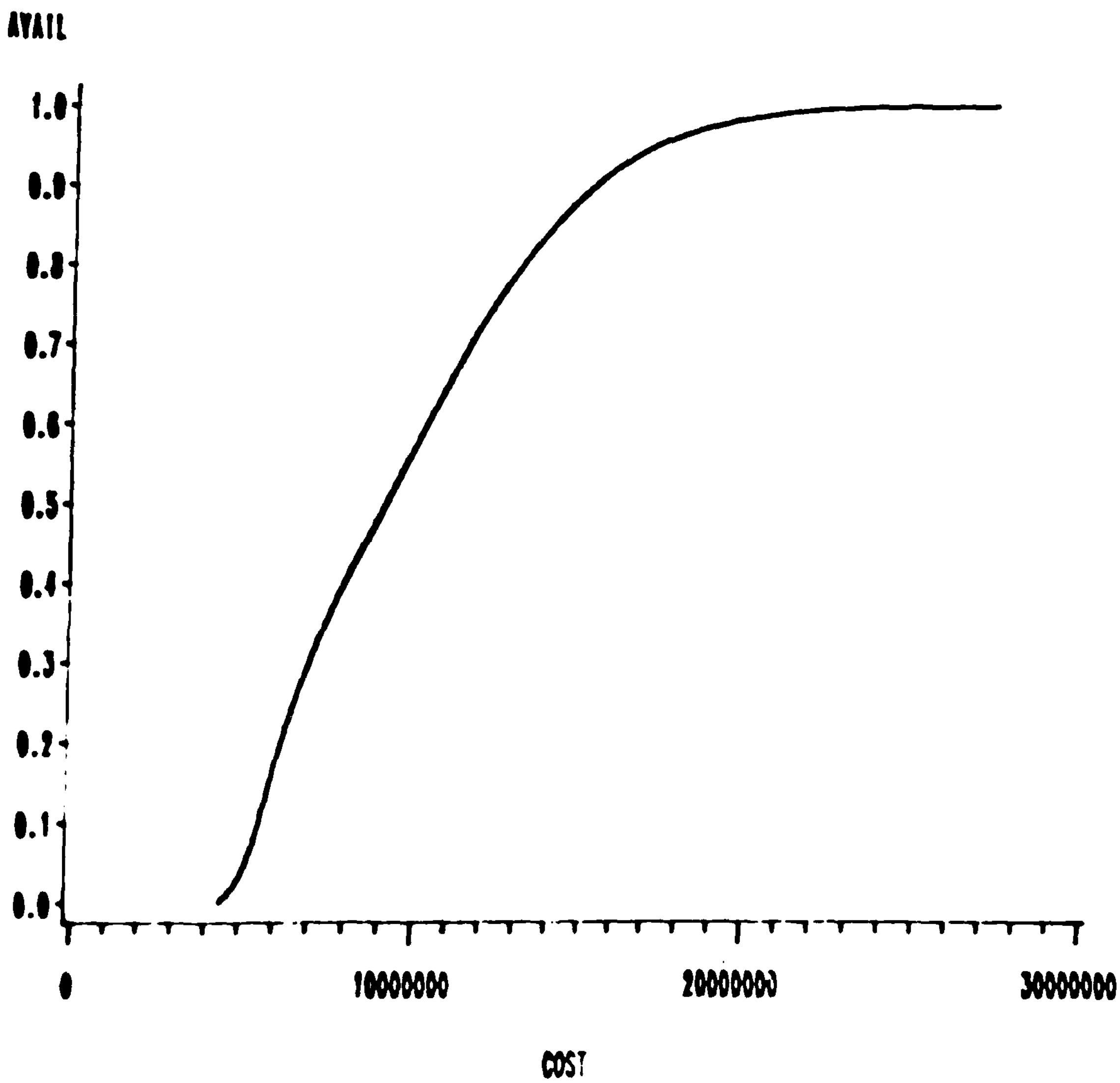


Figure 5. CHIPMUNK - Availability versus Expenditures

BUDGET CODE	EXPENDITURES
1	8 362 843$00
3	84 354$00
4	682 222$00
5	1 860 635$00
8	51 496$00
TOTAL	11 041 550$00

TABLE III. Additional expenditures for baseline scenario.

Analysis of the results. From table III we can recognize the high recommendation for procurement of level 1 components at budget code 1. A total of 8,362,843$00 is the value recommended. Some explanations are found for this high figure. Cannibalization when the aircraft is under maintenance can be one of the explanations for this additional buy though all expenses where already incurred at this time. Since only 30 aircraft are deployed to the base, the other 6 aircraft can be used as spares to increase the actual stocks. This situation suggests that the total assets can be increased by 6 additional units.

The total expenditures with this aircraft type were constant over the last years. Thus the model can be run with this data, in order to predict the total expenditures for fiscal year 1987, and compare the results with the actual expenses incurred in 1986. The results for this run are presented in table IV. These results reflect the consideration of 6 additional units in the total assets for each level 1 component. The total assets for level 2 component does not reflect any buy since we are interested in predicting the total buys for the fiscal year of 1987.

BUDGET CODE	EXPENDITURES
1	332 429$00
4	61 998$00
5	21 469 196$00
8	729 691$00
TOTAL	22 593 314$00

TABLE IV. Total expenditures for fiscal year 1987 for the CHIPMUNK aircraft type

The expenditures recommended by the model for the year of 1987 are equivalent to the total expenses incurred in the replenishment of the stocks for the year of 1986. This can be compared in order to verify the reasonableness of the model.

For the fiscal year of 1986 the total expenses incurred by management area 1 with this aircraft type was 20,000,000$00. Management area 4 spent an amount of money close to 700,000$00. The total expenditures on budget code 1 and 5, codes for management area 1, is 21,801,625$00 and for budget code 4 and 8, management area 2, the total is 791,689$00. The underestimate of the model can be explained by lack of information on the components data file such as the data for level 2 components on base repair pipeline, order and ship pipeline and depot repair pipeline. Although of the missing data, it must be noted that some of these components are not reparable. Anyway the results are close enough to the actual expenses. This close relationship between the model results and the real word situation is the starting point of the acceptance of the model for real applications.

<u>Analysis of sensitivity of the model</u>

Flying hour program and the number of aircraft deployed are fixed independent variables under this model. Either availability rate and total expenditures can be selected to be the third independent variable. For this research availability rate is chosen as the third independent variable and total expenditures selected as the dependent variable.

To study the contribution of each independent variable to the change in total expenditures, two levels were defined for each variable. The baseline scenario is one of them and the other is defined as the values of the variables at baseline scenario plus 10%. The selection of the percentage change of variables normalizes the two levels of comparison. In this way, the contribution of each variable can be compared with the contribution of any other variable. Thus, the three factors (flying hour program, number of units, and availability), at two levels of each, gives a total of 8 treatment combinations. The 8 treatments are analyzed by just running a statistical package that is able to produce an analysis of variance (ANOVA) table with type I sum of squares (TYPE I SS) in order to verify the contribution of each source of variation. The SAS package supported by classroom support computers (CSC) of the Air Force Institute of Technology is used for this analysis. A summary of the ANOVA table is given in table V.

SOURCE	TYPE I SS
HOURS	22323252.34720800
UNITS	304897.71961800
AVAIL	2189229.99776450

TABLE V. Source of variation and TYPE I SS from ANOVA table

From the analysis of the ANOVA table we can conclude that for the baseline scenario with 2400 flying hour program, 30 aircraft and 70% of availability, flying hour program has the greatest contribution to the change of total expenditures. Availability rate is the second variable in contribution to the changes in the independent variable. The number of units deployed has the least contribution to total expenditures change.

Other analyses can be performed with this model. Sometimes the tradeoff between investment in maintenance repair facilities or transportation and the acquisition of spare parts for stocks cannot be easily determined. With this model management can conduct analysis of total expenditure response to the changes in pipelines. As an example, base repair pipeline (BR), order and ship pipeline (OS) and depot pipeline (DR) are analyzed at the baseline scenario at a level of 90% of actual pipelines. The effect of reducing the pipelines by 90%, one at a time, is presented in table VI for both level 1 components and all components.

	TOTAL EXPENDITURES (IN THOUSANDS OF ESCUDOS)	
PIPELINE	LEVEL 1	ALL COMPONENTS
BASELINE	9 129.	11 045.
90% BR	8 896. (97.5%)	10 797. (97.8%)
90% OS	8 694. (95.2%)	10 594. (95.9%)
90% DR	8 631. (94.5%)	10 532. (95.4%)

TABLE VI. Total expenditures for the CHIPMUNK at 90% of the pipelines

From the results presented in table VI. we can conclude that the reduction to 90% in the pipelines always has an effect less then 10%. The decision to invest in reducing the pipelines must consider the total investment required and its benefit in decreasing total expenditures.

Summary

As a conclusion of this chapter, we can state that this model has validity in predicting total expenditures or projecting the availability rates for the CHIPMUNK aircraft type. Flying hour program was found as the independent variable which contributes more to changes in total expenditures, while the number of aircraft has the lowest contribution to these changes. The percentage effect on expenditures of reducing the pipelines is smaller than the percentage reduction in the pipelines. The tradeoff between investment in maintenance facilities or in spare parts procurement can be evaluated with this model. Management can use the model as a tool to decide between different investment alternatives.

V. Conclusions

Summary of the Research

The intent of this research was to develop a model for the Portuguese Air Force which would relate aircraft availability with spare parts, a component of the relationship between aircraft readiness and logistics resources. The motivation for this research came from the increasing concern in the top decision level of the Portuguese Air Force with this subject, and the fact that no adequate tools exist to measure or define this relationship. Since this subject is not new for the USAF, an extensive literature review was conducted in order to have the essential insight required to accomplish this research.

The topic covers a very broad area and this imposed a number of constraints on the research. Also the time frame imposed made it necessary to narrow the scope of the problem. Thus, the research was limited to only one aircraft type deployed at one base. The research objectives were to develop an easy-to-use computer model, which runs quickly and requires a minimum amount of input data. Spare parts was the component of logistics resources which was specifically considered in the modeling. The model determines the availability of a specific aircraft type given a certain level of budget for spare parts and the specified total flying hour program.

The concepts of the supply system and repair cycle were described. A discussion of the METRIC, MOD-METRIC and AAM models was found to be of importance in the model development since these three models provide the necessary framework for the research. This discussion was organized into the following sections: system environment, assumptions, data requirements, model objective and problem solution. Assumptions were formulated in two major groups. One group of the assumptions related to the demand and resupply processes. Another group of assumptions related to the repair process. A description of the context in which the model will be used was presented in order to give a better understanding of why some assumptions were needed. The analysis and discussion in Chapter II leads to the definition of the model to be developed. It is a two-echelon, two-indenture model with multiple items and subject to a budget constraint.

The development of the model was basically oriented by the METRIC, MOD-METRIC and in the majority by the AAM model. The main computational features of the model are the expected backorders technique introduced by METRIC, availability rate and the contribution of each component to the availability of the aircraft, and the optimization procedure, based on the marginal analysis technique. All of these computations were developed in Chapter III under the methodology. Shopping list and cost versus availability curves, were discussed along with the presentation of the algorithms. The development of the mathematical procedures, the structure of the model and the requirements for the model implementation were also described.

In Chapter IV a narrative description of the model was given to explain how the model works and how the user can interact with it. The following step was verification and validation. The selection of the aircraft type to be used and the baseline scenario was defined in order to be able to run the model with real data. The total expenditures projected by the model using this real data compared well with the actual expenses allowing us to begin to validate the model. Analysis of the input data was performed to verify the validity of the values and the relationships given in the data base. This chapter concluded with the analysis of the results and the sensitivity of the model to changes in the independent variables.

Conclusions of the Research Effort

The objectives of this research were met. A model which is easy-to-use, which runs quickly and requires a minimum amount of input data was developed. The model was run for the CHIPMUNK aircraft type with the baseline scenario of a 2400 flying hour program, 30 aircraft, and 70% availability rate. The main conclusions are:

- The model is suitable in predicting total expenditures. The results are close enough to the actual expenses.
- Some data was missing in the input data files. This leads to an underestimation of the expenses.
- Flying hour program is the variable that contributes more to the changes in total expenditures, while the number of units deployed is the variable that contributes the least.
- The reduction of the pipelines to 90% of actual values always has an effect less then 10% on the total expenditures.

Managerial Implications

With the use of this model Portuguese Air Force Managers are able to directly relate funding levels and aircraft readiness. They can allocate money to the needs of different aircraft types by using this model for all aircraft types, one at a time. Tradeoffs can be studied with the model. The investment required to reduce the pipelines and its benefit is one example. However, the final decision must consider the total investment required and its benefit in decreasing total expenditures. Also, and maybe more important than this, managers can optimize the allocation of the money among the spare parts requirements to support each aircraft type. Thus, following the recommendations of the model, the level of funding available under the budget constraint can be spent in the most profitable way.

Suggested Areas of Further Research

The research performed here encountered certain problems. Some problems were diminished through simplification of the approach while others required a great deal of effort to overcome. This research can be the starting point for further work. Some areas of interest in pursuing the research are:

- To extend the model to incorporate several aircraft types.
- To model for manpower and support equipment.
- To model for more than two levels of indenture.
- To include the repair option (i.e., have the repair costs competing for the same funding).
- To explicitly consider common components.

And, may be the most important area to start with, to evaluate the management information system actually in use.

Final Comments

The significance of performing this research and of the conclusions reached lies with the constantly changing environment of the Portuguese Air Force weapon systems and the tightening of budgetary constraints. Management must ensure that every escudo spent achieves maximum mission accomplishment. With the Portuguese Air Force modernization, it is imperative that management be informed of system performance. Availability will remain the most appropriate performance measure for the inventory system.

APPENDIX A

AAM Batch File

This batch file was created for to run the model in the Scientific Support Computer (SSC) using the UNIX system. The file controls the execution of the model up to the generation of the availability versus expenditures curve. The following programs are called by this file:

- HIGHSET
- SAM
- SPLITSRT
- CURVES

AD-A179 514 AIRCRAFT READINESS UNDER BUDGET CONSTRAINT ITS RELATIONSHIP WITH LOGISTIC..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R M LOPES 2/2

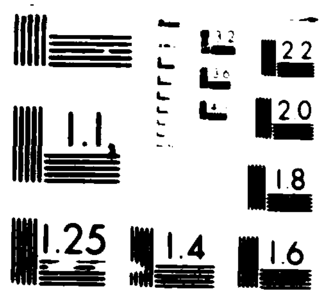
UNCLASSIFIED DEC 86 AFIT/GOR/OS/86D-8 F/G 1/3 NL

END

DATE FILMED

5-87

DTIC



RESOLUTION

```
#
clear
echo ' '
echo '          ****************************************************************     '
echo '          **                                                            **     '
echo '          **                        ATTENTION                           **     '
echo '          **                                                            **     '
echo '          **                                                            **     '
echo '          **   1. DIRECTORY NAMES MUST BE 7-CHARACTER LENGTH.           **     '
echo '          **   2. DATA FILES MUST EXIST  IN ONE SUBDIRECTORY.           **     '
echo '          **                                                            **     '
echo '          ****************************************************************     '
echo ' '
echo ' '
echo ' '
echo '                   If ready hit RETURN to continue ...'
$<
highset
sleep 1
clear
echo ' '
echo '               ***************************************************             '
echo '               **                                               **             '
echo '               **       LEVEL 2 OF THE MODEL IS RUNNING         **             '
echo '               **                                               **             '
echo '               ***************************************************             '
echo '                                                                               '
echo '                      FOR PROCESSING OF SUBASSEMBLIES ...                      '
echo '                                                                               '
echo ' '
sam
DELFILE SORTED 3
echo ' '
echo '                           SORTING OUTPUT RECORDS ...                          '
echo '                                                                               '
echo ' '
splitsrt
mv LEVELNM LEVELNM.OLD
mv LEVELNM.1 LEVELNM
sleep 5
clear
echo ' '
echo '               ***************************************************             '
echo '               **                                               **             '
echo '               **       LEVEL 1 OF THE MODEL IS RUNNING         **             '
echo '               **                                               **             '
echo '               ***************************************************             '
echo '                                                                               '
echo '                       FOR PROCESSING OF ASSEMBLIES ...                        '
echo '                                                                               '
echo ' '
```

```
sam
DELFILE SORTED 2
echo ' '
echo '                    SORTING OUTPUT RECORDS ...                    '
echo '                                                                  '
echo ' '
splitsrt
sleep 5
clear
echo ' '
echo '            **********************************************            '
echo '            **                                          **            '
echo '            **  NOW PRODUCING AVAILABILITY/COST TABLE.  **            '
echo '            **                                          **            '
echo '            **********************************************            '
echo ' '
curves
sleep 1
clear
echo ' '
echo '*****************************************************************************'
echo '**                                                                         **'
echo '**    IF YOU WISH TO RUN ANOTHER MODEL YEAR, ENTER AAM AT THE PROMPT.      **'
echo '**                                                                         **'
echo '**    IF YOU WISH TO RUN THE SHOPPING LIST PROGRAM, ENTER "SHOPLIST"       **'
echo '**                           AT THE PROMPT.                                **'
echo '**                                                                         **'
echo '*****************************************************************************'
```

APPENDIX B

SHOP Batch File

This batch file was created for to run the model in the Scientific Support Computer (SSC) using the UNIX system. The file controls the execution of the model when the shopping list is requested. The following programs are called by this file:

- SHOP1
- SHOP2
- SHOPB

```
#
clear
echo ' '
echo '*************************************************************************'
echo '**                                                                     **'
echo '**                 NOW RUNNING THE SHOPPING LIST PROGRAM               **'
echo '**                                                                     **'
echo '*************************************************************************'
echo ' '
shop1
sort -o $1/SSVPRIME.$21 $1/USVPRIME.$21
mv LEVELNM LEVELNM.BAC
mv LEVELNM.OLD LEVELNM
shop2
shopb
mv LEVELNM LEVELNM.OLD
mv LEVELNM.BAC LEVELNM
sleep 2
clear
echo ' '
echo '*************************************************************************'
echo '**                                                                     **'
echo '**    THE RUNNING OF THE AIRCRAFT AVAILABILITY MODEL IS COMPLETE.      **'
echo '**                                                                     **'
echo '**      IF YOU WISH TO LIST A SHOPPING LIST AT THIS TIME, ENTER        **'
echo '**                                                                     **'
echo '**        "more '$1'/SHOPLIST.'$2'"      FOR ALL COMPONENTS, OR       **'
echo '**        "more '$1'/SHOPBC--.'$2'"      FOR ANY BUDGET CODE,         **'
echo '**                                           WHERE [--] = 1,10.        **'
echo '**                                                                     **'
echo '*************************************************************************'
```

APPENDIX C

Input Data Files

The input data files required for to run the model are:

1. DATA.yy1
2. DATA.yy2
3. HOURS

Legend for DATA.yy1 and DATA.yy2:

Column

1	-	Component Name
2	-	Component Cost
3	-	Base Repair Pipeline
4	-	Order and Ship Pipeline
5	-	Depot Repair Pipeline
6	-	Condemnation Replacement Pipeline
7	-	Total Asset Position
8	-	Name of Next Higher Assembly (NHA)
9	-	Component Fixed Level
10	-	Budget Code

Example of file DATA.yy1

```
EXAMPLEA    2000.00   1.7  0.3  3.6  0.4   12   EXAMPLE  0  1
EXAMPLEB    7000.00   0.9  0.0  2.5  0.1    0   EXAMPLE  0  5
```

Example of file DATA.yy2

```
EXAMPLEAA    500.00   1.2  0.9  5.2  1.5  100   EXAMPLEA 0  1
EXAMPLEAB     10.00   5.1  2.9  3.1  5.0   35   EXAMPLEA 0  1
EXAMPLEBA    200.00   1.2  0.7  2.3  2.1   78   EXAMPLEB 0  5
EXAMPLEBB    100.00   2.4  0.3  4.1  0.0   10   EXAMPLEB 0  5
EXAMPLEBC    900.00   0.0  0.0  0.0  0.0    0   EXAMPLEB 0  5
```

Example of file HOURS

```
2400
```

APPENDIX D

Aircraft Availability Model

FORTRAN Programs and Subroutines

P - Program
S - Subroutine

```
C**** ADDIT,  THE SUBROUTINE THAT COMPUTES SUNK COSTS
      SUBROUTINE ADDIT(N,ITASSE,COST,COM,CUM)
C**** ITASSE WILL BE INCREASED TO N.
C**** DO NOT CALL ADDIT WITH A REAL N.
C**** COM MAY ALREADY BE > 0.,AND CUM INCLUDE THAT.
C**** RESET CUM
      CUM=CUM-COM
C
C**** COMPUTE PROC SUNK COSTS.
      IF(ITASSE.LT.N) COM=COM+COST*(N-ITASSE)
      ITASSE=N
C
C**** NOW SUM THE CUM
      CUM=CUM+COM
      RETURN
      END
```

```
C**** CURVES, THE AVAILABILITY COMPUTATION AND SELECTION PROGRAM
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST,IBUDSRU
      CHARACTER NSN*13
      LOGICAL DEBUG,PRINT
C
      COMMON/SRUCHA/NSNSRU,SONSN
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSREADS,TPARTCOS,APARTCOS
      PARAMETER (ZSRUEBO=2000,ZSUNK=10)
      REAL SVPRIME(0:ZSRUEBO)
      REAL TPARTCOS(ZSUNK), APARTCOS(ZSUNK,0:ZSRUEBO)
      DOUBLE PRECISION SRUEBO(0:ZSRUEBO)
      CHARACTER SONSN*13,NSNSRU*13
C
      INTEGER NUNITS, NBASES
      REAL    VMCONST, PBUYA(5),SUNKC(ZSUNK)
      CHARACTER WSNAME*13, COMMENT*80, DATADIR*7, FNAME*30
      CHARACTER TEST*6

      DEBUG = .FALSE.
      PRINT = .FALSE.
C
C
C**** OPEN INPUT FILES AND READ INITIAL FIGURES
      OPEN (13,FILE='LEVELNM')
      READ (13,*) IYEAR,LEVEL,DATADIR
C
      WRITE (FNAME,6) DATADIR,IYEAR
    6 FORMAT (A,'/WSNATURE.',I2)
      OPEN (1,FILE=FNAME)
      READ (1,*) VMCONST, PBUYA
      READ (1,7) WSNAME,IFLYHRS,NUNITS,NBASES,COMMENT
    7 FORMAT(2X,A,I7,2I4,//A)
      TEST = COMMENT
      NSN = WSNAME
C
      WRITE(FNAME,8)DATADIR,IYEAR
    8 FORMAT(A,'/SORTED.',I2,'1')
      OPEN (3,FILE=FNAME,FORM='UNFORMATTED')
C
      WRITE(FNAME,9)DATADIR,IYEAR
    9 FORMAT(A,'/COSTTOTS.',I2,'1')
      OPEN (9,FILE=FNAME)
      READ(9,*)SUNKC
      TSUNKC=0.
      DO 20 I=1,ZSUNK
         TSUNKC=TSUNKC+SUNKC(I)
         TPARTCOS(I)=SUNKC(I)
```

```
         APARTCOS(I,0)=SUNKC(I)
   20 CONTINUE
C
C
C**** FILL ARRAYS
   27 WRITE(*,*)'                  ENTER "S" TO SAVE AVAILABILITY CURVE'
      WRITE(*,*)'                        OR ENTER <CR> TO CONTINUE'
      READ(*,25)FNAME
   25 FORMAT (A1)
      IF(FNAME.NE.' '.AND.FNAME.NE.'S') GOTO 27
      IF(FNAME.EQ.'S') PRINT =.TRUE.
      WRITE(*,*)'  '
      WRITE(*,*)'  '
   21 WRITE(*,*)'             ENTER COST INCREMENT FOR AVAILABILITY CURVE'
      WRITE(*,*)'              OR ENTER <CR> TO LET PROGRAM CHOOSE VALUE '
      READ(*,22)FNAME
   22 FORMAT(A30)
      WRITE(*,*)'  '
      WRITE(*,*)'  '
      IF(FNAME.EQ.' ')THEN
         COST=.005*TSUNKC
      ELSE
         READ(FNAME,*,ERR=29)COST
         IF(COST.LT.1)COST=.005*TSUNKC
      END IF
      GO TO 30
   29 WRITE(*,*)' INPUT ERROR, PLEASE RE-TYPE'
      GO TO 21
C
   30 CALL SRUSTART
      IF(NSRUEBO.LE.0)GO TO 998
      CALL FSEBO
C
C
C**** FIND END OF PARTCOSA ARRAY
      DO 50 NPART=10,1,-1
         IF(APARTCOS(NPART,NSRUEBO).GT.0.)GO TO 60
   50 CONTINUE
      NPART=1
C
C
C**** PRINT CURVE
   60 WRITE(*,70)WSNAME,IYEAR
   70 FORMAT('  FOR THE "',A13,'"  SYSTEM, FOR THE YEAR 19',I2)
      IF (TEST.NE.'  NONE')  WRITE(*,*)'NOTE: ',COMMENT
      WRITE(*,*)' TOTAL SUNK COSTS = ',TSUNKC
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)'   AVAILABILITY    TOTAL COST'
      WRITE(*,*)' '
C
C**** IF PRINT IS TRUE SAVE CURVE
```

```
      IF(PRINT) THEN
         WRITE(FNAME,690)DATADIR,IYEAR
  690    FORMAT(A,'/CURVE.',I2)
         OPEN (4,FILE=FNAME)
         WRITE(4,70)WSNAME,IYEAR
         IF (TEST.NE.'  NONE') WRITE(4,*)'NOTE: ',COMMENT
         WRITE(4,*)' TOTAL SUNK COSTS = ',TSUNKC
         WRITE(4,*)' '
         WRITE(4,*)' '
         WRITE(4,*)'   AVAILABILITY    TOTAL COST'
         WRITE(4,*)' '
      END IF
C
C     CALCULATE THE POINTS FOR THE CURVE
      DO 100 N=0,NSRUEBO
         IF(SRUEBO(N)/NUNITS.LT.10.)THEN
            E= EXP(-SRUEBO(N) / NUNITS)
         ELSE
            E= 0.
         ENDIF
         NCOST = N*COST+TSUNKC
C-OUT    WRITE(*,*)' '
         WRITE(*,80)E,NCOST
C-OUT&     ,(APARTCOS(I,N),I=1,NPART)
         IF (PRINT) WRITE(4,80)E,NCOST
   80    FORMAT(2X,F13.6,7X,I8)
  100 CONTINUE
      IF (PRINT) CLOSE(4)
C
C
C**** MAKE AND SAVE A DECISION
  110 WRITE(*,*)' '
      WRITE(*,*)' PLEASE ENTER AN AVAILABILITY OR COST OR 0 TO RE-LIST'
      WRITE(*,*)'  THE CURVE. A NUMBER > 100 WILL BE INTERPRETED AS A '
      WRITE(*,*)'   COST, A NUMBER < 100 AS AN AVAILABILITY.'
      WRITE(*,*)' '
      IF (PRINT)
     &WRITE(*,*)'        <><><>   CURVE WILL BE SAVED   <><><>'
      WRITE(*,*)'  '
      WRITE(*,*)'  '
      READ(*,120)FNAME
  120 FORMAT(A30)
      IF(FNAME.EQ.' ')GO TO 60
      READ(FNAME,130,ERR=110)POINT
  130 FORMAT(BN,F10.2)
      IF(POINT.LT.1.E-6)GO TO 60
C
      IF(POINT.GT.100.)THEN
C     ==== INTERPOLATE COST POINT
         DO 200 N=0,NSRUEBO
            TCOST=N*COST+TSUNKC
            IF(TCOST.GT.POINT)GO TO 300
```

```
200     CONTINUE
        WRITE(*,*)' DOLLAR AMOUNT ABOVE TOP OF CURVE.  PLEASE REENTER.'
        GO TO 110
300     IF(N.EQ.0)THEN
          WRITE(*,*)'$$$ AMOUNT BELOW BOTTOM OF CURVE. PLEASE REENTER.'
          GO TO 110
        END IF
        N=N-1
        REMAINDR=POINT-(N*COST+TSUNKC)
        FRACTION=REMAINDR/COST
        EBO=SRUEBO(N)*(1.-FRACTION)+SRUEBO(N+1)*FRACTION
        AVAILOUT=EXP(-EBO/NUNITS)
        COSTOUT=POINT
        SVPOUT=SVPRIME(N)*(1.-FRACTION)+SVPRIME(N+1)*FRACTION
C
      ELSE
C       ==== INTERPOLATE AVAILABILITY POINT. IF IN PERCENT, FIX.
        IF(POINT.GT.1.)POINT=POINT*.01
        EBO=-NUNITS*LOG(POINT)
        DO 400 N=0,NSRUEBO
           IF(EBO.GT.SRUEBO(N))GO TO 500
400     CONTINUE
        WRITE(*,*)' AVAILABILITY ABOVE TOP OF CURVE.  PLEASE REENTER.'
        GO TO 110
500     IF(N.EQ.0)THEN
          WRITE(*,*)'AVAILABILITY BELOW BOTTOM OF CURVE.PLEASE REENTER'
          GO TO 110
        END IF
        N=N-1
        REMAINDR=SRUEBO(N)-EBO
        FRACTION=REMAINDR/(SRUEBO(N)-SRUEBO(N+1))
        AVAILOUT=POINT
        COSTOUT=TSUNKC+COST*(N+FRACTION)
        SVPOUT=SVPRIME(N)*(1.-FRACTION)+SVPRIME(N+1)*FRACTION
      END IF
C
C**** COMPUTE PARTCOSTS FOR DECISION
      DO 600 I=1,ZSUNK
        TPARTCOS(I)=APARTCOS(I,N)*(1.-FRACTION)+APARTCOS(I,N+1)
     &             *FRACTION
600   CONTINUE
C
C**** PRINT DECISION
      WRITE(*,*)'  YOUR SELECTION COMPUTES TO'
      WRITE(*,*)' '
      WRITE(*,*)'  AVAILABILITY     TOTAL COST     SORT VALUE CUTOFF'
      WRITE(*,610)AVAILOUT,COSTOUT,SVPOUT
610   FORMAT(1X,F12.4,6X,E13.6,5X,E14.7)
      WRITE(*,*)' '
      WRITE(*,*)'  THE BREAKOUT OF SUBTOTAL COST BY BUDGET CODE IS'
      WRITE(*,*)' '
      WRITE(*,*)'   BUDGET CODE    COST'
```

```
      WRITE(*,*)' '
      DO 650 I=1,ZSUNK
         WRITE(*,620)I,TPARTCOS(I)
  620    FORMAT(2X,I9,5X,F10.0)
  650 CONTINUE
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)' IF YOU WISH TO REVISE YOUR DECISION ENTER "REDO" '
      WRITE(*,*)'    ELSE ENTER <CR> TO CONTINUE    '
      WRITE(*,*)' '
      READ(*,680)FNAME
  680 FORMAT(A30)
      IF(FNAME.EQ.'REDO'.OR.FNAME.EQ.'redo')GO TO 110
C
C**** OPEN OUTPUT FILE AND SAVE DECISION
      WRITE(FNAME,700)DATADIR,IYEAR
  700 FORMAT(A,'/DECISION.',I2)
      OPEN (5,FILE=FNAME)
      WRITE(5,*)'  YOUR SELECTION COMPUTES TO'
      WRITE(5,*)' '
      WRITE(5,*)'  AVAILABILITY      TOTAL COST      SORT VALUE CUTOFF'
      WRITE(5,610)AVAILOUT,COSTOUT,SVPOUT
      WRITE(5,*)' '
      WRITE(5,*)'  THE BREAKOUT OF SUBTOTAL COST BY BUDGET CODE IS'
      WRITE(5,*)' '
      WRITE(5,*)'   BUDGET CODE    COST'
      WRITE(5,*)' '
      WRITE(5,620)(I,TPARTCOS(I),I=1,ZSUNK)
C
C**** DONE
      WRITE(*,*)' AVAILABILTY CURVE GENERATED'
      STOP
C
C
C**** ERROR EXIT
  998 WRITE(*,*)' '
      WRITE(*,*)'  DATA MALFUNCTION.  NO AVAILABILITY CURVE GENERATED.'
      CALL GETPID(ID)
      CALL KILL(ID,9)
      END
C
C
C
C
C
C*************************************************************
C**** SRUSTART,  STARTS FILLING THE SRUEBO & SVPRIME ARRAYS
      SUBROUTINE SRUSTART
C
C**** THIS SUBROUTINE INITIALIZES THE SRUEBO (& CO.) ARRAYS.
C**** IF NSRUEBO IS RETURNED AS 0. THEN NO CHILDREN.
C**** THIS SUBROUTINE ASSUMES THAT NSN,NSNSRU,SORTV,GLCSRU,
```

```
C**** IBUDSRU, & SONSN ARE DEFINED.
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST,IBUDSRU
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/SRUCHA/NSNSRU,SONSN
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSREADS,TPARTCOS,APARTCOS
      PARAMETER (ZSRUEBO=2000,ZSUNK=10)
      REAL SVPRIME(0:ZSRUEBO)
      REAL TPARTCOS(ZSUNK), APARTCOS(ZSUNK,0:ZSRUEBO)
      DOUBLE PRECISION SRUEBO(0:ZSRUEBO)
      CHARACTER SONSN*13,NSNSRU*13
C
C**** INITIALIZE.
C     NSNSRU='                    '
      NSRUEBO=0
      SRUEBO(0)=0.
C
C**** PROCESS RECORD UNLESS PAST NSN.
      IF (DEBUG) WRITE (*,10) NSNSRU,NSN
   10 FORMAT (' NSNSRU = ',A,' NSN = ',A)
  100 IF(NSNSRU.GT.NSN) RETURN
C
C**** IF A MATCH PROCESS ALL FLAGGED RECORDS.
C
      IF(NSNSRU.EQ.NSN)THEN
C
C         ---- IF A REAL SVSRU IS ENCOUNTERED, YOU'RE DONE.
          IF(SVSRU.LE.500.)GOTO 189
C
C         ---- SUPER LARGE SVSRU IS A FLAG THAT THIS RECORD IS REALLY
C         ----   A STARTING RECORD.  SAVE.  THE GLCSRU IN THESE
C         ----   RECORDS IS ACTUALLY STARTING EBO.  SUM.
C
          SRUEBO(0)=SRUEBO(0)+GLCSRU
          SRUEBO(1)=SRUEBO(0)
          NSRUEBO=1
      END IF
C
C
C**** READ NEXT RECORD AND LOOP BACK.
C
      READ(3,END=199)SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
  150 FORMAT (1X,A13,1X,F18.13,1X,F15.3,1X,A13)
      IF (DEBUG) WRITE(*,*)'record read = ', NSNSRU,SVSRU,GLCSRU,SONSN,
     & IBUDSRU
      NSREADS=NSREADS+1
```

```
      GO TO 100
C
C
C**** SET SVPRIME(0)
  189 SVPRIME(0)=SVSRU*1.0001
      RETURN
C
C
C**** EOF.  SET HIGH NSNSRU TO PREVENT FURTHER READS.
  199 NSNSRU='ZZZZZZZZZZZZZZZZZ'
      RETURN
      END
C
C
C
C
C
C*************************************************************
C     FSEBO,  FILLS THE SRUEBO & SVPRIME ARRAYS.
      SUBROUTINE FSEBO
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST,IBUDSRU
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/SRUCHA/NSNSRU,SONSN
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSREADS,TPARTCOS,APARTCOS
      PARAMETER (ZSRUEBO=2000,ZSUNK=10)
      REAL SVPRIME(0:ZSRUEBO)
      REAL TPARTCOS(ZSUNK), APARTCOS(ZSUNK,0:ZSRUEBO)
      DOUBLE PRECISION SRUEBO(0:ZSRUEBO)
      CHARACTER SONSN*13,NSNSRU*13
C
C
      TSRUCOST=0.
C
C
C**** SINCE SRUSTART BEGAN THE JOB, JUST JUMP RIGHT IN.
C**** SINCE SRUEBO(0) WAS FILLED EARLIER, SOME TESTS ARE UNNECESSARY.
C
      IF (DEBUG) WRITE (*,10) NSNSRU,NSN
   10 FORMAT (' IN FSEBO, NSNSRU = ',A,' NSN = ',A)
  200 IF(NSN.LT.NSNSRU)GO TO 2000
      TSRUCOST=TSRUCOST+GLCSRU
      TPARTCOS(IBUDSRU)=TPARTCOS(IBUDSRU)+GLCSRU
C
C
C**** FILL SRUEBO ARRAY & SVPRIME ARRAY
```

```
C
      IF(NSRUEBO.EQ.ZSRUEBO)GO TO 900
      IF (DEBUG) WRITE(*,*)'COST =',COST
      NLSWORTH=TSRUCOST/COST
C
      IF(NLSWORTH.LT.NSRUEBO)THEN
C
C        ==== SMALL GLUMP.  CONTINUE FILLING A PSWORTH.
C
         SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO)-SVSRU*GLCSRU
C
      ELSE
C        ==== BIG GLUMP HAS FINISHED FILLING AN LSWORTH AND STARTED
C        ====   ON THE NEXT ONE.  FIRST WRAP UP THE FULL ONE.
C
         SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO)-SVSRU*(NSRUEBO*COST-
     &    TSRUCOST+GLCSRU)
  800    SVPRIME(NSRUEBO)=SVSRU
         APARTCOS(IBUDSRU,NSRUEBO) = TPARTCOS(IBUDSRU)-
     &                               (TSRUCOST-COST*NSRUEBO)
C        ==== SAVE PARTCOSTS FOR OTHER BUDGETS TOO.
         DO 810 IBUD=1,ZSUNK
            IF(IBUD.EQ.IBUDSRU)GO TO 810
            APARTCOS(IBUD,NSRUEBO) = TPARTCOS(IBUD)
  810    CONTINUE
C
C        ==== NOW START FILLING NEXT PSWORTH
C        ==== IF CUTOFF HAS NOT BEEN HIT PROCEED.
C        ====================================
C
         IF(NSRUEBO.LT.ZSRUEBO)THEN
C
C           ==== START FILLING NEW PSWORTH BY PUTTING THE REST OF
C           ====  THE GLUMP INTO THE NEW PSWORTH.  UNLESS GLUMP IS
C           ====  SO LARGE AS TO FILL IT ENTIRELY, THEN SWEAT.
C
         NSRUEBO=NSRUEBO+1
            IF(NLSWORTH.LT.NSRUEBO)THEN
C
C              ==== REGULAR CASE.  PUT REST OF GLUMP INTO NEW PSWORTH
C
               SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO-1)-SVSRU*
     &          (TSRUCOST-COST*(NSRUEBO-1))
            ELSE
C
C              ==== GLUMP IS SO LARGE AS TO FILL THIS PSWORTH
C              ====  ENTIRELY.  FILL AND LOOP BACK.
C
               SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO-1)-SVSRU*COST
               GO TO 800
            END IF
C
```

```
C        ==== ELSE   CUTOFF HIT.  TERMINATE.
C
         ELSE
            WRITE(*,850)NSN,SRUEBO(ZSRUEBO)
  850       FORMAT('  <*><*><*><*> NSRUEBO>Z FOR ',A13,'  EBO=',F7.2)
         END IF
C        =====================================
C
      END IF
C
C
C
C**** READ NEXT RECORD AND LOOP BACK TO THE BEGINNING.
C
      IF (DEBUG) WRITE(*,*)' READ A RECORD =',NSREADS
  900 READ(3,END=1999)SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
      IF(DEBUG)WRITE(*,*)'RECORD READ =',NSNSRU,SVSRU,GLCSRU,SONSN
     &        ,IBUDSRU
      NSREADS=NSREADS+1
      IF (DEBUG) WRITE (*,10) NSNSRU,NSN
      GO TO 200
C
C
C
C**** END LOGIC
 1999 NSNSRU='~~~~~~~~~~~~~~~~~~'
 2000 IF(NSRUEBO.GT.0)SVPRIME(NSRUEBO)=0.
      IF(DEBUG)WRITE(*,*)'  SRUEBO ARRAY ',(SRUEBO(I),I=0,NSRUEBO),
     &'  SVPRIME ARRAY  ',(SVPRIME(I),I=0,NSRUEBO)
C
      RETURN
      END
```

```
C**** DFACTLN, THE FUNCTION THAT COMPUTES THE
C****     LOGARITHM (BASE E) OF 'N' FACTORIAL.
C****     IT TAKES AN INTEGER AS INPUT.
C***
      DOUBLE PRECISION FUNCTION DFACTLN(N)
C***
      IMPLICIT DOUBLE PRECISION(D)
      REAL*8 DTABLE(0:30)
C***  *DSIGMA IS A CONSTANT = LN(SQRT(2*PI))
      DATA DSIGMA/.91893 85332 04672 74178D0 /
C***  *DZERO IS THE LOGARITHM (BASE E) OF 0!
C***  *DTABLE(I) IS THE LOGARITHM (BASE E) OF I!
      DATA DTABLE/     0.0D0,
     &                 0.0D0,   .6931471805599453l0D0,
     & .1791759469228055000D1,   .3178053830347945620D1,
     & .4787491742782045990D1,   .6579251212010100990D1,
     & .8525161361065414300D1,   .1060460290274525020D2,
     & .1280182748008146960D2,   .1510441257307551530D2,
     & .1750230784587388580D2,   .1998721449566188620D2,
     & .2255216385312342290D2,   .2519122118273868150D2,
     & .2789927138384089160D2,   .3067186010608067280D2,
     & .3350507345013688890D2,   .3639544520803305360D2,
     & .2933988418719949400D2,   .4233561646075348500D2,
     & .4538013889847690800D2,   .4847118135183522390D2,
     & .5160667556776437360D2,   .5478472939811231920D2,
     & .5800360522297915790D2,   .6126170176100200200D2,
     & .6455753862700633110D2,   .6788974313718153490D2,
     & .7125703896716800900D2,   .7465823634883016430D2
     &/
C***
C***      *IF(N IS WITHIN THE TABLE LIMITS)
           IF((N.LT.0) .OR. (N.GT.30)) GO TO 100
C***
C***         *RETURN TABLE VALUE
              DFACTLN = DTABLE(N)
C***
C***      *ELSE (USE STIRLING'S APPROXIMATION)
           GO TO 200
  100      CONTINUE
C***
C***         *COMPUTE VARIOUS PARTS NEEDED FOR THE APPROXIMATION
              DPN = DBLE(FLOAT(N))
              DFACTLN = (DPN + .5D0)*DLOG(DPN) - DPN + DSIGMA
     &                 + 1.0D0/(12.0D0*DPN)
     &                 - 1.0D0/(360.0D0*DPN*DPN*DPN)
C***
C***      *END IF (TABLE LIMITS TEST)
  200      CONTINUE
C***
      RETURN
      END
```

```
C**** DLNGAMMA,  THE FUNCTION THAT COMPUTES THE NATURAL
C****            LOG OF GAMMA OF DX
C
      DOUBLE PRECISION FUNCTION DLNGAMMA(DX)
C
      IMPLICIT DOUBLE PRECISION(D)
C***  DSIGMA IS A CONSTANT =LN(SQRT(2*PI))
      DATA DSIGMA/.91893 85332 04672 74178 D0/
      IF(DX.LT.10)  WRITE (*,10) DX
   10 FORMAT ('  DLNGAMMA PASSED SMALL DX=',F9.3)
C
C***  COMPUTE VARIOUS PARTS NEEDED FOR THE APPROXIMATION
      DPN=DX-1.D0
      DLNGAMMA = (DPN + .5D0)*DLOG(DPN) - DPN + DSIGMA
     &                 + 1.0D0/(12.0D0*DPN)
     &                 - 1.0D0/(360.0D0*DPN*DPN*DPN)
C
       RETURN
       END
```

```
C**** FEBO, THE SUBROUTINE THAT FILLS THE EBO ARRAY
      SUBROUTINE FEBO
C
C**** THIS SUBROUTINE FILLS THE EBO ARRAY.  EBO(IEBO) IS THE
C**** EBO FOR IEBO'S L'SWORTHS OF INVESTMENT IN THE COMPONENT.
C**** IEBO=IS-ISRUSTRT+(ID+IB-ITASSE) WHERE IS IS THE NUMBER OF
C**** L'SWORTHS OF SRU'S, ID IS THE NUMBER AT THE DEPOT, IB IS
C**** THE NUMBER AT THE BASES, & ITASSE IS THE STARTING ASSET
C**** POSITION AFTER SACROSANCT BUYS.
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/EBOBLK/EBO,NSRU,NEBO,NBASES,ITASSE,BOPIPE
     & ,DPIPE,CUTOFF,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ
     & ,DPIPOVRQ,EBOS,MAXREP,ALPHA,MEBO,INKR,NSKIPPED,NDONE
      PARAMETER (ZEBO=1000)
      DIMENSION EBO(0:ZEBO),NSRU(0:ZEBO)
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C
      IF(DEBUG)WRITE(*,*)'-->IN FEBO: ISRUSTRT[IN]=',ISRUSTRT
   10 FORMAT(A)
C**** INITIALIZE
      DO 100 IEBO=0,NEBO
         EBO(IEBO)=99999.
  100 CONTINUE
      NEBO=0
      NSRU(0)=ISRUSTRT
C
C
C**** OUTER LOOP (ID).
C     ****************************************************************
      MAXIM=ITASSE+ZEBO*INKR
      IF (DEBUG) WRITE(*,*)'--> IN FEBO: LUMPD,MAX,INKR ',
     &                                   LUMPD,MAXIM,INKR
      DO 700 ID=LUMPD,ITASSE+ZEBO*INKR,INKR
         DVAR=DE2BO-DEBO*DEBO
C
C
C        ----------------------------------------
         NS=MIN(NSRUEBO,ITASSE+ZEBO*INKR-ID)
         DO 500 IS=ISRUSTRT,NS
```

```
C
C          ---- IF THIS (ID,IS) COMBINATION IS BEATEN BY (ID+1,IS-1)
C          ----- THEN EXIT IS LOOP (GO TO NEXT ID).
C
           IF(IS.GT.ISRUSTRT.AND.SRUEBO(IS-1)-SRUEBO(IS).LE.DREBO)
     &        THEN
              NSKIPPED=NSKIPPED+NS+1-IS
              GO TO 600
C
C          ----- IF THIS (ID,IS) COMBINATION IS BEATEN BY (ID-1,IS+1)
C          -----   THEN EXIT IS LOOP (GO TO NEXT IS) UNLESS TASSE IS
C          -----   A CONSTRAINT
C
           ELSEIF(ID.GT.LUMPD.AND.IS.LT.NS
     &        .AND.SRUEBO(IS)-SRUEBO(IS+1).GT.DREBO+DTERM)THEN
C
C             ----- IF ID IS PAST TASSE SKIP, ELSE DO PART OF NB LOOP.
C
              IF(ID.GT.ITASSE)THEN
                 NSKIPPED=NSKIPPED+1
                 GO TO 500
              ELSE
                 NB=MIN(ITASSE-ID , ZEBO*INKR+ITASSE-ID-IS+ISRUSTRT)
              END IF
           ELSE
              NB=ZEBO*INKR+ITASSE-ID-IS+ISRUSTRT
           END IF
C
           NDONE=NDONE+1
C
C          ----- SET UP FOR BASE LOOP.
C
           B1PIPE=(DEBO+BOPIPE+SRUEBO(IS))/NBASES
           CALL LUMPCMP(B1PIPE,Q
     &      ,LUMPB1,B1EBO,B1REBO,B1TERM,B1E2BO,B1QM1OVQ,B1PIPOVQ)
           IB1=LUMPB1
C
           LUMPB=NBASES*LUMPB1
           IF(NB.LT.LUMPB)GO TO 600
C
C          ----- IB1 LOOP. NOTE; IB = NBASES*IB1.
C          ------------------------------------------
           DO 400 IB=LUMPB,NB,NBASES
C
C             ----- IF # LRUs GE TASSE & SYSEBO A NEW WINNER, SAVE.
C
              IEBOLD=IS+IB+ID-ITASSE-ISRUSTRT
              IEBO=(IEBOLD+INKR-1)/INKR
              SYSEBO=NBASES*B1EBO
              IF(IB+ID.GE.ITASSE.AND.SYSEBO.LT.EBO(IEBO))THEN
                 EBO(IEBO)=SYSEBO
                 NSRU(IEBO)=IS
```

```
C
C---------------------------------------------------------------
                END IF
C
C               ==== CHECK EBO(0) AND UPDATE AS NECC.
C
               IF(IS.EQ.ISRUSTRT.AND.IEBOLD.GT.-NBASES.AND.IEBOLD.LT.0)
     &          THEN
                   SYS0EBO=SYSEBO+IEBOLD*B1REBO
                   IF(SYS0EBO.LT.EBO(0))EBO(0)=SYS0EBO
               END IF
C
C               ==== COMPUTE NEXT EBO,REBO,ETC. EXIT LOOP IF REBO<CUTOFF.
C
                B1EBO=B1EBO-B1REBO
                IF(B1REBO.LT.CUTOFF.OR.B1EBO.LT.0.)GO TO 450
                B1E2BO=B1E2BO-B1EBO-B1EBO-B1REBO
                B1TERM=B1TERM*(B1PIPOVQ+B1QM1OVQ*IB1)/(IB1+1)
                B1REBO=B1REBO-B1TERM
                IB1=IB1+1
  400        CONTINUE
C            =======================
C
  450     IF(IEBO.GT.NEBO)NEBO=IEBO
  500     CONTINUE
C         ========================================
C
C         ==== COMPUTE NEXT DEBO,DREBO,ETC.  EXIT LOOP IF REBO < CUTOFF.
C
  600     DO 650 IID=ID,ID+INKR-1
             DEBO=DEBO-DREBO
             IF(DREBO.LT.CUTOFF.OR.DEBO.LT.0.)GO TO 800
             DE2BO=DE2BO-DEBO-DEBO-DREBO
             DTERM=DTERM*(DPIPOVRQ+DQM1OVRQ*IID)/(IID+1)
             DREBO=DREBO-DTERM
  650     CONTINUE
C
  700 CONTINUE
C     *************************************************************
C
C
  800 IF (DEBUG) WRITE(*,*)'—> LEAVING FEBO'
      RETURN
      END
```

```
C     FSEBO,  FILLS THE SRUEBO & SVPRIME ARRAYS.
      SUBROUTINE FSEBO
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C
      TSRUCOST=0.
C
C
C
C*********************************************************************
C*********************************************************************
C**** SINCE SRUSTART BEGAN THE JOB, JUST JUMP RIGHT IN.
C**** SINCE SRUEBO(0) WAS FILLED EARLIER, SOME TESTS ARE UNNECESSARY.
C
      IF (DEBUG) WRITE (*,10) NSNSRU,NSN
   10 FORMAT (' IN FSEBO, NSNSRU = ',A,' NSN = ',A)
  200 IF(NSN.LT.NSNSRU)GO TO 2000
      TSRUCOST=TSRUCOST+GLCSRU
C
C
C**** FILL SRUEBO ARRAY & SVPRIME ARRAY
C
      IF(NSRUEBO.EQ.ZSRUEBO)GO TO 900
      NLSWORTH=TSRUCOST/COST
C
C     ***************************************************************
      IF(NLSWORTH.LT.NSRUEBO)THEN
C
C        ==== SMALL GLUMP.  CONTINUE FILLING A PSWORTH.
C
         SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO)-SVSRU*GLCSRU
C
C     ****
      ELSE
C        ==== BIG GLUMP HAS FINISHED FILLING AN LSWORTH AND STARTED
C        ====   ON THE NEXT ONE.  FIRST WRAP UP THE FULL ONE.
C
         SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO)-SVSRU*(NSRUEBO*COST-
     &    TSRUCOST+GLCSRU)
  800    SVPRIME(NSRUEBO)=SVSRU
```

```
C
C        ==== NOW START FILLING NEXT PSWORTH
C        ==== IF CUTOFF HAS NOT BEEN HIT PROCEED.
C        ========================================
C
         IF(NSRUEBO.LT.ZSRUEBO)THEN
C
C           ==== START FILLING NEW PSWORTH BY PUTTING THE REST OF
C           ===>  THE GLUMP INTO THE NEW PSWORTH.  UNLESS GLUMP IS
C           ====  SO LARGE AS TO FILL IT ENTIRELY, THEN SWEAT.
C
         NSRUEBO=NSRUEBO+1
            IF(NLSWORTH.LT.NSRUEBO)THEN
C
C              ==== REGULAR CASE.  PUT REST OF GLUMP INTO NEW PSWORTH
C
               SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO-1)-SVSRU*
     &          (TSRUCOST-COST*(NSRUEBO-1))
            ELSE
C
C              ==== GLUMP IS SO LARGE AS TO FILL THIS PSWORTH
C              ====  ENTIRELY.  FILL AND LOOP BACK.
C
               SRUEBO(NSRUEBO)=SRUEBO(NSRUEBO-1)-SVSRU*COST
               GO TO 800
            END IF
C
C        ==== ELSE   CUTOFF HIT.  TERMINATE.
C
         ELSE
            WRITE(*,850)NSN,SRUEBO(ZSRUEBO)
  850       FORMAT('  <*><*><*><*> NSRUEBO>Z FOR ',A13,'  EBO=',F7.2)
         END IF
C        ========================================
C
      END IF
C     ************************************************************
C
C
C
C**** READ NEXT RECORD AND LOOP BACK TO THE BEGINNING.
C
  900 READ(3,END=1999)SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
      IF (DEBUG) WRITE (*,910) SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
  910 FORMAT (' IN FSEBO ',A13,1X,E13.7,1X,F13.2,1X,A13,I3)
      NSREADS=NSREADS+1
      GO TO 200
C*********************************************************************
C*********************************************************************
C
C
C
```

```
C
C**** END LOGIC
 1999 NSNSRU='~~~~~~~~~~~~~~~~~~'
 2000 IF(NSRUEBO.GT.0)SVPRIME(NSRUEBO)=0.
      IF(DEBUG)WRITE(*,*)'  SRUEBO ARRAY ',(SRUEBO(I),I=0,NSRUEBO),
     &'  SVPRIME ARRAY  ',(SVPRIME(I),I=0,NSRUEBO)
      RETURN
      END
```

```
C**** FSV, SUBROUTINE THAT FILLS THE SV ARRAY (CONVEXIFICATION)
      SUBROUTINE FSV
C**** THIS SUBROUTINE READS EBO(0) THROUGH EBO(NEBO) & NSRU AND
C**** LOADS THE CONVEXIFIED OUTPUT INTO SV AND NSRUCON & GLCOST
C**** STARTING AT ISV(1)=(EBO(1)-EBO(0))/COST
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/EBOBLK/EBO,NSRU,NEBO,NBASES,ITASSE,BOPIPE
     & ,DPIPE,CUTOFF,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ
     & ,DPIPOVRQ,EBOS,MAXREP,ALPHA,MEBO,INKR,NSKIPPED,NDONE
      PARAMETER (ZEBO=1000)
      DIMENSION EBO(0:ZEBO),NSRU(0:ZEBO)
C
      COMMON/SVBLK/SV,GLCOST,NSRUCON,NSV,LASTREP,MSV
      PARAMETER (ZSV=1000)
      DIMENSION SV(0:ZSV),GLCOST(0:ZSV),NSRUCON(0:ZSV)
C
C**** COMPUTE INCR SO THAT ZSV WON'T BE A CONSTRAINT
      INCR=MAX(1,1+NEBO/ZSV)
      IF(INCR.GT.1)THEN
         DEBUG=.TRUE.
         WRITE(*,25)NSN,INCR
   25    FORMAT('0<*><*><*>INCR > 1 FOR ',A13,'  INCR=',I4)
      END IF
      IEBOLAST=0
C
C*************************************************************
      DO 2000 IEBO=1,NEBO,INCR
         IF(EBO(IEBO).GE.EBO(IEBOLAST))GO TO 2000
         IF(NSV.GE.ZSV)GO TO 3000
C        ==== COMPUTE VALUES FOR NEXT ENTRY
         NSV=NSV+1
         GLCOST(NSV)=COST*(IEBO-IEBOLAST)*INKR
         SV(NSV)=(EBO(IEBOLAST)-EBO(IEBO))/GLCOST(NSV)
         NSRUCON(NSV)=NSRU(IEBO)
         IEBOLAST=IEBO
C
C        ==== CHECK FOR CONVEXITY OF LAST ENTRY
 1000    IF(SV(NSV).LT.SV(NSV-1))GO TO 2000
C           === LAST ENTRY NOT CONVEX.  MERGE LAST TWO ENTRIES
C           === TO MAKE A NEW LAST ENTRY, THEN CHECK NEW MERGED
C           === ENTRY FOR CONVEXITY.
            NSVTEMP=NSV
            NSV=NSV-1
            SUM=SV(NSV)*GLCOST(NSV)+SV(NSVTEMP)*GLCOST(NSVTEMP)
            GLCOST(NSV)=GLCOST(NSV)+GLCOST(NSVTEMP)
```

```
            SV(NSV)=SUM/GLCOST(NSV)
            NSRUCON(NSV)=NSRUCON(NSVTEMP)
            IF(NSV.EQ.LASTREP)LASTREP=LASTREP-1
         GO TO 1000
C
 2000 CONTINUE
C*************************************************************
C
      GO TO 4000
C
 3000 WRITE(*,*)' <*><*>ZSV EXCEEDED ON ',NSN,'NEBO=',NEBO
 4000 RETURN
      END
```

```
C     HIGHSET
C
C     HIGHSET IS A PROGRAM WHICH DOES THE PREREQUISITE INITIALIZATION
C     FOR THE RUNNING OF THE AVAILABILITY MODEL. THIS PROGRAM SAVES THE
C     NUMBER OF THE LEVEL OF THE RUN IN FILE 'LEVELNM', AND ESTABLISHES
C     VARIOUS RUN PARAMETERS IN FILE 'WSNATURE'.
C
C
C
      INTEGER NUNITS, NBASES

      REAL    VMCONST, PIPEBUYP, PBUYA(5), SUNKCS(60)
C
      CHARACTER WSNAME*13, COMMENT*80, DATADIR*7, FNAME*30
      CHARACTER TEMP*10, NSN*13, WRK*20
C
      DO 121 I=1,25
         WRITE(*,*)' '
  121 CONTINUE
C
    1 WRITE(*,2)
    2 FORMAT(/,'   ENTER THE NAME OF THE DIRECTORY TO BE USED'
     & ,/,'       FOR THE INPUT AND OUTPUT DATA FILES '
     & ,/,'    OR ENTER <CR> TO USE DIRECTORY "AAM.OUT" ')
      READ(*,101)DATADIR
  101 FORMAT(A7)
      IF(DATADIR.EQ.' ')THEN
         DATADIR = 'AAM.OUT'
         WRITE(*,102)DATADIR
  102    FORMAT('&',A7)
      END IF
    4 WRITE(*,5)
    5 FORMAT(/,'   ENTER THE BUDGET YEAR (TWO DIGITS)    ')
      READ(*,6)TEMP
    6 FORMAT(A4)
      READ(TEMP,8,ERR=4)IYEAR
    8 FORMAT(BN,I5)
      IF(IYEAR.LT.80.OR.IYEAR.GT.99)THEN
         WRITE(*,*)' YEAR MUST BE BETWEEN 80 AND 99.  PLEASE RETYPE.'
         GO TO 4
      END IF
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)' IF YOU WISH TO REVISE YOUR DECISION ENTER "REDO" '
      WRITE(*,*)'   ELSE ENTER <CR> TO CONTINUE   '
      WRITE(*,*)' '
      READ(*,6)TEMP
      IF(TEMP.EQ.'REDO'.OR.TEMP.EQ.'redo')GO TO 1
C
C
C     OPEN THE 'LEVELNM' FILES AND WRITE THE LEVEL (& CO.) TO THEM.
C
```

```
      OPEN (13,FILE='LEVELNM')
      OPEN (14,FILE='LEVELNM.1')
      LEVEL = 2
      WRITE (13,7) IYEAR,LEVEL,DATADIR
    7 FORMAT(2I3,1X,A)
      LEVEL = 1
      WRITE (14,7) IYEAR,LEVEL,DATADIR
C
C
C-----------------------------------------------------------------
C
C     COLLECT THE PARAMETERS FOR THIS RUN AND WRITE THEM TO A FILE
C

      WRITE (FNAME,10) DATADIR,IYEAR
   10 FORMAT (A,'/WSNATURE.',I2)
      OPEN (7,FILE=FNAME,ERR=970)
C
C
C     OBTAIN THE CONSTANT FOR THE VMR.
C
   11 WRITE (*,12)
   12 FORMAT (/'    ENTER THE VARIANCE TO MEAN RATIO.'/
     &          ' OR ENTER <CR> TO USE THE DEFAULT VALUE OF 1.0    ')
C
      READ(*,13)TEMP
   13 FORMAT(A10)
      IF(TEMP.EQ.'<CR>'.OR.TEMP.EQ.' ')THEN
         VMCONST=1.0
         WRITE(*,113)VMCONST
  113    FORMAT('&',F4.1)
      ELSE
         READ(TEMP,14,ERR=11)VMCONST
   14    FORMAT(F3.1)
      END IF
      IF   (VMCONST .LE. 0.0 )    VMCONST = 1.0
C
C
C     OBTAIN THE VALUE FOR THE PERCENTAGE OF THE PIPELINE TO BUY.
C
   19 WRITE (*,20)
   20 FORMAT (/' ENTER THE FRACTION OF THE PIPELINE',
     & ' WHICH IS TO BE BOUGHT,'/
     & ' OR ENTER <CR> TO USE THE DEFAULT VALUE OF 1.0, WHICH',/,
     & ' BUYS THE WHOLE PIPELINE.    ')
C
      READ(*,25)TEMP
   25 FORMAT(A10)
      IF(TEMP.EQ.' ')THEN
         PIPEBUYP = 1.0
         WRITE(*,26)PIPEBUYP
   26    FORMAT('&',F4.1)
```

```
      ELSE
         READ(TEMP,14,ERR=30)PIPEBUYP
      END IF
      IF(PIPEBUYP.GE.0.)GO TO 50
   30 WRITE(*,40)
   40 FORMAT(' INPUT ERROR, PLEASE RETYPE.')
      GO TO 19
   50 WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)' IF YOU WISH TO REVISE YOUR DECISION ENTER "REDO" '
      WRITE(*,*)'   ELSE ENTER <CR> TO CONTINUE  '
      WRITE(*,*)' '
      READ(*,6)TEMP
      IF(TEMP.EQ.'REDO'.OR.TEMP.EQ.'redo')GO TO 11
C
      DO 100 I= 1,5
         PBUYA(I) = PIPEBUYP
  100 CONTINUE
C
      WRITE (7,*) VMCONST, PBUYA
C
C---------------------------------------------------------------
C
C     OBTAIN WEAPON SYSTEM CHARACTERISTICS
C
  200 WRITE (*,300)
  300 FORMAT (/' ENTER THE NAME OF THE SYSTEM - UP TO 13 CHARACTERS.')
C
      READ (*,400) WSNAME
  400 FORMAT (A13)
      IF(WSNAME.EQ.' ')THEN
         WRITE(*,*)' YOU MUST ENTER A SYSTEM NAME'
         GO TO 200
      END IF
C
  420 WRITE (*,430) WSNAME
  430 FORMAT(/' ENTER THE FLYING HOUR PROGRAM FOR SYSTEM "',A13,'"')
      READ(*,6)TEMP
      IF(TEMP.EQ.' ')THEN
         WRITE(*,*)' YOU MUST ENTER A NUMBER GREATER THAN 0'
         GO TO 420
      ELSE
         READ(TEMP,8,ERR=420)IFLYHRS
         FLYHRS=IFLYHRS
      END IF
C
  450 WRITE (*,500) WSNAME
  500 FORMAT (/' ENTER THE NUMBER OF UNITS OF  "',A13,'"  DEPLOYED ')
      READ(*,6)TEMP
      IF(TEMP.EQ.' ')THEN
         WRITE(*,*)' YOU MUST ENTER A NUMBER GREATER THAN 0'
         GO TO 450
```

```
      ELSE
         READ(TEMP,8,ERR=450)NUNITS
      END IF
C
  550 WRITE (*,600)
  600 FORMAT (/' ENTER THE NUMBER OF BASES  ')
      READ(*,6)TEMP
      IF(TEMP.EQ.' ')THEN
         WRITE(*,*)' YOU MUST ENTER A NUMBER GREATER THAN 0'
         GO TO 550
      ELSE
         READ(TEMP,8,ERR=550)NBASES
      END IF
C
      WRITE (*,700)
  700 FORMAT (/' ENTER ANY ONE LINE OF COMMENTS REGARDING THIS ',/
     & '    MODEL RUN, OR ENTER <CR> FOR NO COMMENT'/)
      READ (*,800)  COMMENT
  800 FORMAT (A80)
      IF(COMMENT.EQ.' ')COMMENT='NONE'
      WRITE(*,*)' '
      WRITE(*,*)' '
      WRITE(*,*)' IF YOU WISH TO REVISE YOUR DECISION ENTER "REDO" '
      WRITE(*,*)'    ELSE ENTER <CR> TO CONTINUE    '
      WRITE(*,*)' '
      READ(*,6)TEMP
      IF(TEMP.EQ.'REDO'.OR.TEMP.EQ.'redo')GO TO 200
C
C
C**** WRITE IT TO THE FILE
      WRITE (7,810) WSNAME,IFLYHRS,NUNITS,NBASES
  810 FORMAT(2X,A,I7,2I4)
      WRITE (7,*) COMMENT
C
C
C**** UPDATE THE COMPONENT DATA FILES
      CALL UPDATA(DATADIR,IYEAR,FLYHRS)
C
C
C**** BUILD DATADIR/COSTTOTS.YY3
      WRITE(FNAME,820)DATADIR,IYEAR
  820 FORMAT(A,'/COSTTOTS.',I2,'3')
      OPEN(9,FILE=FNAME)
      WRITE(9,*)SUNKCS
C
C
C**** BUILD DATADIR/SORTED.YY3
      WRITE(FNAME,840)DATADIR,IYEAR
  840 FORMAT(A,'/SORTED.',I2,'3')
      OPEN (3,FILE=FNAME,FORM='UNFORMATTED')
      IBUDCODE=1
      NSN = '~~~~~~~~~~~~~~~~~~'
```

```
      ANUM=0.0
      WRITE(3) NSN,ANUM,ANUM,NSN,IBUDCODE
C
C
C**** BUILD DELFILE
      OPEN(15,FILE='DELFILE')
      WRITE(15,900)DATADIR,IYEAR
  900 FORMAT('rm ',A,'/$1.',I2,'$2')
      WRK = 'chmod 755 DELFILE'
      CALL SYSTEM(WRK)
C
C
C**** BUILD SHOPLIST
      OPEN(17,FILE='SHOPLIST')
      WRITE(17,950)DATADIR,IYEAR
  950 FORMAT('SHOP ',A,I3)
      WRK = 'chmod 755 SHOPLIST'
      CALL SYSTEM(WRK)
      GOTO 999
C
C
C**** ERROR ON DIRECTORY NAME
  970 WRITE(*,*)'DIRECTORY "',DATADIR,'" NOT FOUND.  PLEASE CHECK.'
      CALL GETPID(ID)
      CALL KILL(ID,9)
C
C
  999 CONTINUE
      WRITE(*,*)' INITIALIZATION DONE'
      END
```

```
C**** LUMPCMP,  THE SUBROUTIN THAT INITIALIZES THE EBO COMPUTATION
      SUBROUTINE LUMPCMP(PIPE,Q,
     &                   LUMP,EBO,REBO,TERM,E2BO,QM1OVERQ,PIPEOVRQ)
C
C       EBO IS COMPUTED AT A CLAIMANT (I.E. A BASE OR DEPOT).  PIPE
C       IS THE EXPECTED # IN RESUPPLY AT THE CLAIMANT, AND Q IS THE
C       VARIANCE-TO-MEAN RATIO.  LUMP IS THE NUMBER OF SPARES GIVING
C       AN EBO REDUCTION OF 1.  LUMP = IS TAKEN AS INPUT.
C       WHEN THE PIPELINE IS BIG ENOUGH, LUMP IS NON-ZERO, AND WHEN
C       THIS HAPPENS LUMP SPARES ARE ALLOCATED SACROSANCT.  EBO IS
C       THE EBO WITH LUMP SPARES.  REBO IS THE EBO REDUCTION FOR THE
C       (LUMP+1)'TH SPARE. TERM IS THE PROB. OF EXACTLY LUMP UNITS
C       IN RESUPPLY.  TO COMPUTE TERM WE USE ISTART=MAX(ZERO OR PIPE-6
C       STD DEVS.),  WHICH GETS US FAR ENOUGH OUT INTO THE LEFT 'TAIL'
C       TO BE ZERO AS FAR AS THE COMPUTER CAN TELL.
C       E2BO IS THE E(BO SQUARED). I.E. THE SECOND MOMENT.
C       QM1OVERQ=(Q-1)/Q.  PIPEOVRQ=PIPE/Q.  ISTART IS WHERE
C       TERM ~= 10**-10 SO THAT 1.-TERM <1 IN SINGLE PREC.
C
C
C &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
C &         ***** PROGRAM LOGIC *****                             &
C &                                                               &
C &         IF  (LUMP<0)  SET LUMP=0                              &
C &                                                               &
C &              IF (POISSON)  COMPUTE APPROPRIATELY              &
C &         ELSE (NBD)                 DITTO                      &
C &              END IF                                           &
C &                                                               &
C &         ELSE (LUMP IS GREATER THAN 0)                         &
C &                                                               &
C &              IF (ISTART=0)                                    &
C &                                                               &
C &                   IF (POISSON)  COMPUTE APPROPRIATELY         &
C &                ELSE (NBD)              DITTO                  &
C &                   END IF                                      &
C &                                                               &
C &              ELSE (ISTART>0)                                  &
C &                                                               &
C &                  IF (POISSON)  COMPUTE APPROP.(USING DFACTLN  &
C &                  ELSE (NBD)     AND DLNGAMMA FUNCTIONS FOR    &
C &                  END IF         NAT LOG OF N FACTORIAL AND    &
C &                                 GAMMA FUNCTION RESPECTIVELY)  &
C &                                                               &
C &              END IF                                           &
C &                                                               &
C &              NOW ITERATE FROM ISTART TO LUMP TO               &
C &                COMPUTE EBO WITH LUMP SPARES.                  &
C &                                                               &
C &        END IF                                                 &
C &&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
C
```

```
C
C
      IMPLICIT DOUBLE PRECISION (D)
C
      LUMP=PIPE-3.*SQRT(PIPE*Q)
C
      IF(LUMP.LE.0)THEN
C        === LUMP=0.  SIMPLE PROCESSING.
         LUMP=0
         EBO=PIPE
C
         IF(Q.LE.1.0001)THEN
            Q=1.
            TERM=EXP(-PIPE)
         ELSE
            TERM=Q**(-PIPE/(Q-1.))
         END IF
         REBO=1.-TERM
         E2BO=Q*PIPE+EBO*EBO
         QM1OVERQ=(Q-1.)/Q
         PIPEOVRQ=PIPE/Q
C
C
C     ****
      ELSE
C        === LUMP > 0.  MAY GET STICKY.
         ISTART = PIPE - 6*SQRT(PIPE*Q)
C
C
C        === COMPUTE P(ISTART), I.E. TERM.
C        ==================================
         IF(ISTART.LE.0)THEN
C           === NOT TOO BAD.  NO ISTART.
            ISTART=0
            IF(Q.LE.1.0001)THEN
               Q=1.
               TERM=EXP(-PIPE)
            ELSE
               TERM=Q**(-PIPE/(Q-1.))
            END IF
C
C        ===
         ELSE
C           === ISTART > 0.  GETTING TOUGH.
C           ===============
            IF(Q.LE.1.0001)THEN
               Q=1.
               DPIPE=PIPE
               TRMLOG=SNGL(-DPIPE+ISTART*DLOG(DPIPE)-
     &                                DFACTLN(ISTART))
            ELSE
               DQ=Q
```

```
              DPIPE=PIPE
              DPOVRQM1=DPIPE/(DQ-1.D0)
              DQM1OVRQ=(DQ-1.D0)/DQ
              TRMLOG=SNGL((-DPOVRQM1)*DLOG(DQ)+ISTART*
     &          DLOG(DQM1OVRQ)+DLNGAMMA(DPOVRQM1+ISTART)-
     &                DFACTLN(ISTART)-DLNGAMMA(DPOVRQM1))
            END IF
C           ================
C
            TERM=EXP(TRMLOG)
         END IF
C        ========================================
C
C
C        ==== NOW ITERATE FROM ISTART TO LUMP.
         EBO=PIPE-ISTART
         REBO=1.-TERM
         E2BO=PIPE*Q+EBO*EBO
         QM1OVERQ=(Q-1.)/Q
         PIPEOVRQ=PIPE/Q
C
         DO 100 I=ISTART+1,LUMP
            EBO=EBO-REBO
            E2BO=E2BO-EBO-EBO-REBO
            TERM=TERM*(QM1OVERQ*(I-1)+PIPEOVRQ)/I
            REBO=REBO-TERM
  100    CONTINUE
C
      END IF
C     ********************************************************
C
      RETURN
      END
```

```
C**** MARG, THE SUBROUTINE WHICH CONTROLS MARGINAL ANALYSIS
      SUBROUTINE MARG
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/EBOBLK/EBO,NSRU,NEBO,NBASES,ITASSE,BOPIPE
     & ,DPIPE,CUTOFF,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ
     & ,DPIPOVRQ,EBOS,MAXREP,ALPHA,MEBO,INKR,NSKIPPED,NDONE
      PARAMETER (ZEBO=1000)
      DIMENSION EBO(0:ZEBO),NSRU(0:ZEBO)
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C
      COMMON/SUNKBLK/COMINS,CUMINS,COMNAS,CUMNAS,COMNEG
     & ,CUMNEG,COMPIP,CUMPIP,COMSRU,CUMSRU,IBUDCODE
      PARAMETER (ZSUNK=10)
      REAL CUMINS(ZSUNK),CUMNAS(ZSUNK),CUMNEG(ZSUNK)
      REAL CUMPIP(ZSUNK),CUMSRU(ZSUNK)
C
      COMMON/SVBLK/SV,GLCOST,NSRUCN,NSV,LASTREP,MSV
      PARAMETER (ZSV=1000)
      DIMENSION SV(0:ZSV),GLCOST(0:ZSV),NSRUCN(0:ZSV)
C
   10 FORMAT(A)
C
C
C
C*******************************************************************
C** 0000000000000000000000  INITIALIZE  0000000000000000000000000 **
C*******************************************************************
C**** SET CUTOFF
      CUTOFF=MIN(MAX(COST*0.00000003,0.0003),0.05)
C**** Q AND MSIZED WILL BE USED IN SOME LATER COMPS
      DVAR=Q*DPIPE
      MSIZED=DPIPE+6.*SQRT(DVAR)
C
      IF(DPIPE.GT.0..AND.NBASES.GT.1)THEN
C
         CALL LUMPCMP(DPIPE,Q
     &   ,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ,DPIPOVRQ)
C
      ELSE
```

```
         DEBO=DPIPE
         LUMPD=0
         DTERM=0.
         DREBO=0.
         DE2BO=DPIPE*DPIPE+Q*DPIPE
      END IF
C
C**** IF THERE WAS AN SRUEBO MATCH, FILL SRUEBO ARRAYS.
C**** IF NOT THE NSRUEBO PARAMETERS WILL BE ZERO.  TEST.
C
      IF(NSRUEBO.GT.0) CALL FSEBO
C
C
C
C*****************************************************************
C** 00000000000000  PROCUREMENT ONLY PROCESSING  00000000000000 **
C*****************************************************************
C**** SET CUTOFF FOR PROCUREMENT
  700 CUTOFF=MIN(MAX(COST*0.00000002,0.0003),0.05)
C
C**** COMPUTE ISRUSTRT
      PTHIS=BOPIPE+SRUEBO(0)
      DVAR=DE2BO-DEBO*DEBO
      B1PIPE=(PTHIS+DEBO)/NBASES
      B1VAR=Q*B1PIPE
      BUMPCHK=FLOAT(ITASSE-LUMPD)/FLOAT(NBASES)
      DELTACHK=1./NBASES
      BUMPSAVE=B1PIPE-3.*SQRT(B1VAR)
C
      IF(NSRUEBO.EQ.0)GO TO 780
      DO 775 ISRUSTRT=0,NSRUEBO
         IF(BUMPSAVE.LT.BUMPCHK)GO TO 790
         PTHIS=BOPIPE+SRUEBO(ISRUSTRT+1)
         B1PIPE=(PTHIS+DEBO)/NBASES
         B1VAR=Q*B1PIPE
         BUMP=B1PIPE-3.*SQRT(B1VAR)
         IF((BUMPSAVE-BUMP).LT.DELTACHK)GO TO 790
         BUMPSAVE=BUMP
  775 CONTINUE
  780 ISRUSTRT=NSRUEBO
      IF(DEBUG)WRITE(*,*)'-->IN MARG: ISRUSTRT=NSRUEBO ',ISRUSTRT
C
C**** SRU SUNK COSTS COUNT.  PAY UP.
  790 IF(ISRUSTRT.GT.0)WRITE(*,*)' NSN,ISRUSTRT = ',NSN,ISRUSTRT
      COMSRU=COST*(ISRUSTRT)
      CUMSRU(IBUDCODE)=CUMSRU(IBUDCODE)+COMSRU
      LUMPB1=MAX(0.,BUMPSAVE)
      IF(ITASSE.LT.LUMPD+NBASES*LUMPB1)CALL ADDIT
     & (LUMPD+NBASES*LUMPB1,ITASSE,COST,COMPIP,CUMPIP(IBUDCODE))
C
C**** COMPUTE INKR SO THAT ZEBO WON'T BE A CONSTRAINT
      PTHIS=BOPIPE+SRUEBO(ISRUSTRT)
```

```
      B1PIPE=PTHIS/NBASES
      B1VAR=Q*B1PIPE
      MSIZEB1=B1PIPE+6.*SQRT(B1VAR)
      MSIZE=MSIZED+NBASES*MSIZEB1-ITASSE
      INKR=1+MSIZE/ZEBO
C     IF(INKR.NE.1)WRITE(*,*)' NSN = ',NSN,' INCR ',INKR
C
C**** FILL EBO ARRAY
      IF(DEBUG)WRITE(*,*)'-->IN MARG CALLING FEBO ISRUSTRT= '
     &                   ,ISRUSTRT
      CALL FEBO
      IF(DEBUG)WRITE(*,*)'EBO ARRAY & CO '
     &                              ,(EBO(I),NSRU(I),I=0,NEBO)
      EBOS=EBO(0)
      NSRUCN(0)=NSRU(0)
      NSV=0
      IF(NEBO.EQ.0)RETURN
      IF(NEBO.GT.MEBO)MEBO=NEBO
C**** CONVEXIFY
      CALL FSV
      IF(DEBUG)WRITE(*,*)' SV ARRAY ',(SV(I),I=0,NSV)
      IF(NSV.GT.MSV)MSV=NSV
      RETURN
      END
```

```
C     PRECFILL,  THE SUBROUTINE WHICH INITIALIZES THE VARIABLES
      SUBROUTINE PRECFILL(C)
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/EBOBLK/EBO,NSRU,NEBO,NBASES,ITASSE,BOPIPE
     & ,DPIPE,CUTOFF,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ
     & ,DPIPOVRQ,EBOS,MAXREP,ALPHA,MEBO,INKR,NSKIPPED,NDONE
      PARAMETER (ZEBO=1000)
      DIMENSION EBO(0:ZEBO),NSRU(0:ZEBO)
C     DOUBLE PRECISION EBO
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C     DOUBLE PRECISION SVSRU,GLCSRU,RGLCSRU
C
      COMMON/SUNKBLK/COMINS,CUMINS,COMNAS,CUMNAS,COMNEG
     & ,CUMNEG,COMPIP,CUMPIP,COMSRU,CUMSRU,IBUDCODE
      PARAMETER (ZSUNK=10)
      REAL CUMINS(ZSUNK),CUMNAS(ZSUNK),CUMNEG(ZSUNK)
      REAL CUMPIP(ZSUNK),CUMSRU(ZSUNK),TSUNKC(ZSUNK)
C
      COMMON/SVBLK/SV,GLCOST,NSRUCN,NSV,LASTREP,MSV
      PARAMETER (ZSV=1000)
      DIMENSION SV(0:ZSV),GLCOST(0:ZSV),NSRUCN(0:ZSV)
C     DOUBLE PRECISION SV,GLCOST
C
C
      CHARACTER C*1
C
C
C
      MEBO = 0
      NSREADS = 0
      MSV = 0
      DO 100 I=0,ZEBO
         EBO(I) = 0.0
         NSRU(I) = 0
  100 CONTINUE
      DO 120 I=0,ZSRUEBO
         SRUEBO(I) = 0.0
         SVPRIME(I) = 0.0
  120 CONTINUE
```

```
      DO 140 I=1,ZSONSNT
         SONSNT(I) = C
  140 CONTINUE
      DO 160 I=1,ZSUNK
         CUMINS(I) = 0.
         CUMNAS(I) = 0.
         CUMNEG(I) = 0.
         CUMPIP(I) = 0.
         CUMSRU(I) = 0.
         TSUNKC(I) = 0.
  160 CONTINUE
      DO 180 I=0,ZSV
         SV(I) = 0.0
         GLCOST(I) = 0.0
         NSRUCN(I) = 0
  180 CONTINUE
C
C
      RETURN
      END
```

```
C**** SAM,  MAIN PROGRAM FOR SUBASSEMBLIES
C
C
C
C
C
C                FILE CODES USED WITHIN THIS PROGRAM :
C              (WHERE n INDICATES THE LEVEL OF THE RUN.)
C     FILE
C     CODE     FILENAME        DESCRIPTION
C
C      1       RESULTS.n       MAIN OUTPUT OF MODEL RESULTS
C      2       UNSORTED.n      UNSORTED RECORDS
C      3       SORTED.n+1      SORTED RECORDS
C      7       NATURE          CONTAINS WEAPON SYSTEM CHARACTERISTICS
C      8       COSTTOTS.n+1    COST TOTALS WRITTEN IN THE PREVIOUS
C                              LEVEL RUN
C      9       COSTTOTS.n      COST TOTALS WRITTEN IN THIS LEVEL RUN
C     11       COMPDATA.n      THE UPDATED COMPONENT DATA FILE
C     13       LEVELNM         CONTAINS THE LEVEL NUMBER FOR THE RUN
C
C
C
C
C
C
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/EBOBLK/EBO,NSRU,NEBO,NBASES,ITASSE,BOPIPE
     & ,DPIPE,CUTOFF,LUMPD,DEBO,DREBO,DTERM,DE2BO,DQM1OVRQ
     & ,DPIPOVRQ,EBOS,MAXREP,ALPHA,MEBO,INKR,NSKIPPED,NDONE
      PARAMETER (ZEBO=1000)
      DIMENSION EBO(0:ZEBO),NSRU(0:ZEBO)
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C
      COMMON/SUNKBLK/COMINS,CUMINS,COMNAS,CUMNAS,COMNEG,CUMNEG
     &                ,COMPIP,CUMPIP,COMSRU,CUMSRU,IBUDCODE
      PARAMETER (ZSUNK=10)
      REAL CUMINS(ZSUNK),CUMNAS(ZSUNK),CUMNEG(ZSUNK)
      REAL CUMPIP(ZSUNK),CUMSRU(ZSUNK),TSUNKC(ZSUNK)
```

```
C
      COMMON/SVBLK/SV,GLCOST,NSRUCN,NSV,LASTREP,MSV
      PARAMETER (ZSV=1000)
      DIMENSION SV(0:ZSV),GLCOST(0:ZSV),NSRUCN(0:ZSV)
C
C
      CHARACTER WSNAME*13,DATADIR*7
      CHARACTER NHANSN*13,NSNIN*13,FNAME*30
      INTEGER NUNITS,NBASES
      REAL PBUYA(5)
C
C
C
C*********************************************************************
C** 0000000000  INITIALIZE  AND  OPEN  FILES  00000000000000000   **
C*********************************************************************
C
      DEBUG=.FALSE.
      CALL PRECFILL ('0')
      NSNSRU='             '
      SV(0)=999.
      SVPRIME(0)=999.
      NEBO=ZEBO
C
C**** OPEN LEVELNM FILE AND READ IN THE YEAR, LEVEL & DIRECTORY.
      OPEN (13,FILE='LEVELNM')
      READ (13,*)  IYEAR,LEVELNM,DATADIR
      IF (DEBUG) WRITE(*,*)' IY,LVL,DATA=',IYEAR,LEVELNM,' ',DATADIR
C
C**** OPEN FILE 7, THE NATURE FILE, WHICH CONTAINS THE CHARACTERISTICS
C****   OF THE WEAPON SYSTEM WHICH HAVE BEEN INPUT BY THE USER.
      WRITE (FNAME,5) DATADIR,IYEAR
    5 FORMAT (A,'/WSNATURE.',I2)
      OPEN (7,FILE=FNAME)
C
C**** OPEN FILE 1, THE MAIN FILE FOR MODEL RESULTS
      WRITE (FNAME,10) DATADIR,IYEAR,LEVELNM
   10 FORMAT (A,'/RESULTS.',I2,I1)
      OPEN (1,FILE=FNAME,FORM='UNFORMATTED')
C
C**** OPEN FILE 2, THE UNSORTED RECORDS FILE
      WRITE (FNAME,20) DATADIR,IYEAR,LEVELNM
   20 FORMAT (A,'/UNSORTED.',I2,I1)
      OPEN (2,FILE=FNAME,FORM='UNFORMATTED')
C
C**** OPEN FILE 3, THE SORTED RECORDS FILE (FROM PREVIOUS RUN)
      LP1 = LEVELNM + 1
      WRITE (FNAME,30) DATADIR,IYEAR,LP1
   30 FORMAT (A,'/SORTED.',I2,I1)
      OPEN (3,FILE=FNAME,FORM='UNFORMATTED')
C
C**** OPEN FILE 8, THE COST TOTALS FROM THE PREVIOUS LEVELS RUN
```

```
      WRITE (FNAME,40) DATADIR,IYEAR,LP1
   40 FORMAT (A,'/COSTTOTS.',I2,I1)
      OPEN (8,FILE=FNAME)
C
C**** OPEN FILE 9, THE COST TOTALS FILE TO BE WRITTEN IN THIS RUN
      WRITE (FNAME,40) DATADIR,IYEAR,LEVELNM
      OPEN (9,FILE=FNAME)
C
C**** OPEN FILE 11, THE COMPONENT DATA FILE
      WRITE (FNAME,50) DATADIR,IYEAR,LEVELNM
   50 FORMAT (A,'/COMPDATA.',I2,I1)
      OPEN (11,FILE=FNAME,FORM='UNFORMATTED')
C
C**** READ PARAMETERS
      READ(7,*)Q,PBUYA
      PBUY=PBUYA(LEVELNM)
      IF (DEBUG) WRITE(*,*)'  Q,PBUY ',Q,PBUY
C
C**** READ WEAPON SYSTEM NAME, FLYHOURS, # OF AIRCRAFTS, # OF BASES
      READ(7,55)WSNAME,IFLYHRS,NUNITS,NBASES
   55 FORMAT(2X,A,I7,2I4)
      WRITE(*,60)LEVELNM,WSNAME,IFLYHRS,NBASES
   60 FORMAT('  ANALYZING LEVEL',I2,' COMPONENTS INSTALLED ON ',A13
     &       ,/,'  FOR ',I5,' FLYING HOUR PROGRAM AND'
     &       ,/,'  USED AT ONE DEPOT AND',I2,' BASE(S)')
C
C**** READ COST TOTALS FROM LEVEL BELOW.
      READ (8,*) TSUNKC,CUMINS,CUMNAS,CUMNEG,CUMPIP,CUMSRU
C
C
C
C
C***********************************************************************
C**  000000  BEGIN NEW COMPONENT - INITIALIZE VARIABLES  0000000   **
C***********************************************************************
  100 COMINS=0.
      COMNAS=0.
      COMNEG=0.
      COMPIP=0.
      COMSRU=0.
C
C
C**** READ COMPONENT DATA - PIPELINES ETC
      READ(11,END=999)NSNIN,COST,BRPIPE,OSPIPE,DRPIPE,CONPIPE
     &,ITASSE,NHANSN,NEGLEV,IBUDCODE
      NIN=NIN+1
      IF(MOD(NIN,1000).EQ.0)WRITE(*,150)NIN
  150 FORMAT('  NO. OF NSNS PROCESSED=',I6)
      IF(COST.LE.0.)GO TO 100
      NSN=NSNIN
      IF(BRPIPE+OSPIPE.LT.0.)WRITE(*,*)' B+O<0. FOR NSN ',NSNIN
      IF(DRPIPE.LT.0.)WRITE(*,*)'  DRPIPE<0. FOR NSN ',NSNIN
```

```
C
C
C
C*********************************************************************
C**  000000000000  COMPUTE SUNK $ & TEST FOR MARG  0000000000000   **
C*********************************************************************
C
      NPROCESS=NPROCESS+1
C
C**** COMPUTE ADDITIVES AND ADJUST ITASSE
      BOPIPE=BRPIPE+OSPIPE
      DPIPE=DRPIPE+CONPIPE
      ROTA=DPIPE+BOPIPE
      IF(DEBUG)WRITE(*,*)' ITASSE,BOPIPE,DPIPE = ',ITASSE,BOPIPE,' '
     &         ,DPIPE
      IF(ITASSE.LT.0)CALL ADDIT(0,ITASSE,COST,COMNAS,CUMNAS(IBUDCODE))
      IF(NEGLEV.GT.ITASSE) CALL ADDIT(NEGLEV,ITASSE,COST,COMNEG,
     &          CUMNEG(IBUDCODE))
      IPIPE=PBUY*ROTA
      IF(ITASSE.LT.IPIPE) CALL ADDIT(IPIPE,ITASSE,COST,COMPIP,
     &  CUMPIP(IBUDCODE))
C
C**** ELSE TEST FOR MARG, FIRST START FILLING SRUEBO ARRAYS AND CO.
C
      CALL SRUSTART
C
C**** IF ITASSE NOT > PIPE +5*SQRT(PIPE) MARG NECC.
      DVAR=Q*DPIPE
      MSIZED=DPIPE+5.*SQRT(DVAR)
      PTHIS=BOPIPE+SRUEBO(0)
      B1PIPE=PTHIS/NBASES
      B1VAR=Q*B1PIPE
C
      MSIZEB1=B1PIPE+5.*SQRT(B1VAR)
      ITASTEST=MSIZED+NBASES*MAX(1,MSIZEB1)
      IF(DEBUG)WRITE(*,*)' ITASSE,ITASTEST = ',ITASSE,' ',ITASTEST
      IF(ITASSE.LE.ITASTEST)GO TO 800
C
C
C
C
C*********************************************************************
C**  0000000000000000  OUTPUT FOR NO MARG BUYS  0000000000000000    **
C*********************************************************************
  700 SUNKC=COMINS+COMNAS+COMNEG+COMPIP+COMSRU
      IF(SUNKC.GT.10000000.)THEN
         DEBUG=.TRUE.
         WRITE(*,*)' <*><*><*><*> SUNKC=',SUNKC
      END IF
      IF(DEBUG)WRITE(*,*)'  SUNKC=',SUNKC
C**** IF NO $ SPENT DON'T BOTHER TO WRITE.
      IF(SUNKC.LE.0.)GO TO 100
```

```
C
      SVPSTART=SVPRIME(NSRUCN(0))
      WRITE(1)NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,COMNEG,
     & COMPIP,EBOS,SVPSTART,NSONSNT,NHANSN,0,IBUDCODE
  710 FORMAT(A13,1X,F10.2,1X,I8,3(1X,E13.6)/4(E13.6,1X),I4,1X,
     &        A13/I5,1X,I3)
C
      IF(NSONSNT.GT.0)THEN
         WRITE(1)(SONSNT(I),I=1,NSONSNT)
  720    FORMAT(4(1X,A13))
         IW1T=IW1T+1
      END IF
      TSUNKC(IBUDCODE)=TSUNKC(IBUDCODE)+SUNKC
      IW1=IW1+1
      IW1T=IW1T+1
      GO TO 100
C
C
C
C
C*********************************************************************
C**  0000000000000  MARGINAL ANALYSIS PROCESSING  000000000000   **
C*********************************************************************
C**** COMPUTE MISCELLANEOUS QUANTITIES FOR MA.
  800 NEEDMA=NEEDMA+1
C**** CALL MARGINAL ANALYSIS SUBROUTINE
C****   THE MARGINAL ANALYSIS SUBROUTINE WILL IN TURN CALL
C****   OTHER SUBROUTINES AND WHEN FINISHED WILL RETURN THE
C****   CONVEXIFIED ARRAYS READY FOR WRITING TO FILE 1 AND FILE 2.
      IF(DEBUG)WRITE(*,*)'  CALLING MARG FROM SAM'
      CALL MARG
C**** IF MARG MADE NO MARGINAL BUYS GO TO NO MARG BUYS OUTPUT.
      IF(NSV.LE.0)GO TO 700
C
C
C
C
C*********************************************************************
C**  00000000000000  OUTPUT WITH MARGINAL BUYS  00000000000000   **
C*********************************************************************
      SUNKC=COMINS+COMNEG+COMNAS+COMPIP+COMSRU
      IF(SUNKC.GT.10000000.)THEN
         DEBUG=.TRUE.
         WRITE(*,*)' <*><*><*><*> SUNKC=',SUNKC
      END IF
      IF(DEBUG)WRITE(*,*)'  SUNKC=',SUNKC
      SVPSTART=SVPRIME(NSRUCN(0))
      WRITE(1)NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,COMNEG,
     & COMPIP,EBOS,SVPSTART,NSONSNT,NHANSN,NSV,IBUDCODE
C
      DUM1 = 999.
      WRITE (2) NSN,DUM1,EBOS,NHANSN,IBUDCODE
```

```
      IW1=IW1+1
      IW1T=IW1T+NSV+1
      IW2T=IW2T+NSV+1
      IF(NSONSNT.GT.0)THEN
         WRITE(1)(SONSNT(I),I=1,NSONSNT)
         IW1T=IW1T+1
      END IF
      TSUNKC(IBUDCODE)=TSUNKC(IBUDCODE)+SUNKC
C
C**** WRITE MARGINAL GLUMPS
      TDOL=COMSRU
      DO 950 I=1,NSV
         TDOL=TDOL+GLCOST(I)
         NBOUGHT=0.5+TDOL/COST
         NLRUS=NBOUGHT-NSRUCN(I)
         SVP=SVPRIME(NSRUCN(I))
         WRITE(1)SV(I),GLCOST(I),NLRUS,SVP
  930    FORMAT(2E14.7,I8,1X,E14.7)
         WRITE(2)NSN,SV(I),GLCOST(I),NHANSN,IBUDCODE
  950 CONTINUE
      GO TO 100
C
C
C
C*********************************************************************
C** 0000000000000000000000    WRAP UP    0000000000000000000000    **
C*********************************************************************
C
C     WRITE A REC TO INSURE AGAINST EMPTY FILES.
C
  999 CLOSE(11,STATUS='DELETE')
      NSN = '~~~~~~~~~~~~~~~~~~'
      ANUM=0.0
      WRITE(2) NSN,ANUM,ANUM,NSN,IBUDCODE
      IW2T=IW2T+1
C
      IF (DEBUG) THEN
         WRITE(*,1000)NIN,NEEDMA
 1000    FORMAT(' # OF COMPONENTS PROCESSED',I5,' # WITH A REQ=',I6)
         WRITE(*,1029)IW1,IW1T
 1029    FORMAT('  NSN''S TO RESULTS FILE',I5
     &          ,'  RESULTS FILE RECORDS',I12)
         WRITE(*,1030)IW2T,IYEAR,LEVELNM
 1030    FORMAT(I6,' RECORDS WRITTEN TO UNSORTED.',I2,I1)
         WRITE(*,1031)NPROCESS,NSREADS
 1031    FORMAT(' NPROCESS=',I5,' NSREADS=',I8)
         WRITE(*,*)' MEBO,MSV=',MEBO,MSV
      END IF
      WRITE(9,*)TSUNKC,CUMINS,CUMNAS,CUMNEG,CUMPIP,CUMSRU
      WRITE(*,1050)NIN,IW1
 1050 FORMAT('  NO. OF COMPONENTS PROCESSED=',I4/
     & '  NO. WITH POTENTIAL EXPENDITURES=',I4)
```

```
C
      WRITE(*,*)' ANALYSIS COMPLETED'
      END
```

```
C**** SHOP1, SHOPPING LIST PROGRAM FOR LEVEL 1 COMPONENTS
C**** SHOP GENERATES SHOPPING LIST FROM RESULTS FILE
C
      IMPLICIT INTEGER(Z)
      PARAMETER (ZSONSNT=300)
      CHARACTER NSN*13,NHANSN*13,SONSNT(ZSONSNT)*13,FNAME*30
      CHARACTER DATADIR*7,WSNAME*13
      LOGICAL DEBUG
      REAL MINSV,PBUYA(5)
C
C
      DEBUG=.FALSE.
      IF(DEBUG)WRITE(*,*)'  OPEN LEVELNM    '
C
C**** READ DIRECTORY NAME & YEAR FROM LEVELNM FILE
      OPEN (13,FILE='LEVELNM')
      READ (13,*) IYEAR,LEVELNM,DATADIR
C
      WRITE (FNAME,6) DATADIR,IYEAR
    6 FORMAT (A,'/WSNATURE.',I2)
      OPEN (11,FILE=FNAME)
      READ (11,*) VMCONST, PBUYA
      READ (11,8) WSNAME,IFLYHRS,NUNITS,NBASES
    8 FORMAT(2X,A,I7,2I4)
C
      IF(DEBUG)WRITE(*,*)'  OPEN RESULTS'
C**** OPEN FILE 1, THE RESULTS FILE FOR MODEL RESULTS
      WRITE (FNAME,10) DATADIR,IYEAR,LEVELNM
   10 FORMAT (A,'/RESULTS.',I2,I1)
      OPEN (1,FILE=FNAME,FORM='UNFORMATTED')
C
C
      IF(DEBUG)WRITE(*,*)'  OPEN USVPRIME '
C**** OPEN FILE 2, THE SVPRIME FILE TO BE WRITTEN
C                    TO THE NEXT LEVEL
      WRITE (FNAME,20) DATADIR,IYEAR,LEVELNM
  20  FORMAT (A,'/USVPRIME.',I2,I1)
      OPEN (2,FILE=FNAME)
C
C
      IF(DEBUG)WRITE(*,*)'  OPEN DECISION'
C**** OPEN FILE 3, UNLIKE THE SRU SHOP PROGRAM,
C                    THIS WILL JUST CONTAIN 1 #.
      WRITE(FNAME,22)DATADIR,IYEAR
   22 FORMAT(A,'/DECISION.',I2)
      OPEN (3,FILE=FNAME)
      DO 19 I=1,2
         READ(3,*) FNAME
   19 CONTINUE
      READ(3,*)AVAILOUT,COSTOUT,SVCUTOFF
      IF(DEBUG)WRITE(*,*)'AVAILOUT,COSTOUT,SVCUTOFF= '
     &                    ,AVAILOUT,COSTOUT,SVCUTOFF
```

```
C
C
      IF(DEBUG)WRITE(*,*)'  OPEN SHOPLIST'
C**** OPEN FILE 7, THE OUTPUT SHOPPING LIST
      WRITE (FNAME,25) DATADIR,IYEAR,LEVELNM
   25 FORMAT (A,'/SHOPLIST.',I2,I1)
      OPEN (7,FILE=FNAME,FORM='UNFORMATTED')
C
C
C**** DON'T OPEN TCOST FILE FROM THE PREVIOUS LEVELS RUN,
C                                 THERE WASN'T ANY.
      TCOST=0
C
C
      IF(DEBUG)WRITE(*,*)'  OPEN TCOST '
C**** OPEN TCOST FILE TO BE WRITTEN IN THIS RUN
      WRITE (FNAME,40) DATADIR,IYEAR,LEVELNM
  40  FORMAT (A,'/TCOST.',I2,I1)
      OPEN (9,FILE=FNAME,FORM='UNFORMATTED')
C
C
C****************************************************************
C**** BEGIN NEW COMPONENT, READ RECORD FROM RESULTS FILE    ****
C****************************************************************
C
      IF(DEBUG)WRITE(*,*)'  READ FILE 1 '
   50 READ(1,END=999)NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,
     & COMNEG,COMPIP,EBOS,SVPSTART,NSONSNT,NHANSN,NSV,IBUDCODE
   60 FORMAT(A13,1X,F10.2,1X,I8,3(1X,E13.6)/4(E13.6,1X),I4,1X,
     &       A13/I5,1X,I3)
      NREAD=NREAD+1
C
      IF(DEBUG)WRITE(*,*)'  TAPE 1 HEADER   ',
     & NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,COMNEG,
     & COMPIP,EBOS,RVPSTART,NSONSNT,NHANSN,NSV,IBUDCODE
C
      IF(NSONSNT.GT.0)THEN
         READ(1)(SONSNT(I),I=1,NSONSNT)
   70    FORMAT(4(1X,A13))
         IF(DEBUG)WRITE(*,*)' SONSNT ',(SONSNT(I),I=1,NSONSNT)
      END IF
      SUNKC=COMINS+COMNAS+COMNEG+COMPIP
      NPROC=SUNKC/COST+.5
      IF(DEBUG)WRITE(*,*)' NPROC,ITASSE=',NPROC,ITASSE
      MINSV=1000.
      SVPBAWT=SVPSTART
C
C
C**** NOTHING TO READ, THE CUTOFF IS THE SAME FOR ALL COMPONENTS
      MINSV=SVCUTOFF
C
C**** PROCESS COMPONENT, BUYING DOWN THE MARGINAL RECORDS
```

```
C****  UNTIL MINSV IS HIT
  110 LRUSBAWT=0
      IF(NSV.LE.0)GO TO 150
      DO 120 I=1,NSV
            READ(1)SV,GLCOST,NLRUS,SVP
            IF(DEBUG)WRITE(*,*)'  REC IS ',SV,GLCOST,NLRUS,SVP
            IF(SV.LT.MINSV)GO TO 130
            SVPBAWT=SVP
            LRUSBAWT=NLRUS
  120 CONTINUE
      GO TO 150
C
  130 IF(I.EQ.NSV)GO TO 150
      DO 140 J=1+I,NSV
         READ(1)SV
  140 CONTINUE
C
C**** COMPUTE FINAL VALUES AND WRITE
  150 IF(NSONSNT.GT.0)THEN
         DO 155 I=1,NSONSNT
            WRITE(2,152)SONSNT(I),SVPBAWT
  152       FORMAT(A13,E14.7)
  155    CONTINUE
      END IF
      ITARGET=ITASSE+LRUSBAWT
      NLRUSPRO=NPROC+LRUSBAWT
      IF(NLRUSPRO.EQ.0)GO TO 50
      IF(DEBUG)WRITE(*,*)'  ITARGET,NLRUSPRO=',ITARGET,NLRUSPRO
      TCOST=TCOST+COST*NLRUSPRO
      NWRITES=NWRITES+1
      WRITE(7)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
      IF(DEBUG)
     &   WRITE(*,160)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
  160 FORMAT(1X,A13,F10.2,4I7)
      GO TO 50
C
C
C*****************************************************************
C**********    WRAP UP.  WRITE TOTALS TO COST FILE    **********
C*****************************************************************
  999 WRITE(9)TCOST
      IF (DEBUG) WRITE(*,*)'  NWRITES=',NWRITES,'  NREAD =',NREAD
C
      WRITE(*,*)'  TOTAL EXPENDITURES FOR LEVEL 1 =',TCOST
      WRITE(2,152)'~~~~~~~~~~~~~~~~~',0.
      WRITE(*,*)' '
      END
```

```
C**** SHOP2, SHOPPING LIST PROGRAM FOR LEVEL 2 COMPONENTS
C**** SHOP GENERATES SHOPPING LIST FROM RESULTS FILE
C
      IMPLICIT INTEGER(Z)
      PARAMETER (ZSONSNT=300)
      CHARACTER NSN*13,NSNIN*13,NHANSN*13,SONSNT(ZSONSNT)*13
      CHARACTER FNAME*30,DATADIR*7,WSNAME*13
      LOGICAL DEBUG
      REAL MINSV,PBUYA(5)
      NSNIN='             '
      DEBUG =.FALSE.
C
C
C**** READ DIRECTORY NAME & YEAR FROM LEVELNM FILE
      OPEN (13,FILE='LEVELNM')
      READ (13,*) IYEAR,LEVELNM,DATADIR
C
      WRITE (FNAME,6) DATADIR,IYEAR
    6 FORMAT (A,'/WSNATURE.',I2)
      OPEN (11,FILE=FNAME)
      READ (11,*) VMCONST, PBUYA
      READ (11,8) WSNAME,IFLYHRS,NUNITS,NBASES
    8 FORMAT(2X,A,I7,2I4)
C
C
C**** OPEN FILE 1, THE RERULTS FILE FOR MODEL RESULTS
      WRITE (FNAME,10) DATADIR,IYEAR,LEVELNM
   10 FORMAT (A,'/RESULTS.',I2,I1)
      OPEN (1,FILE=FNAME,FORM='UNFORMATTED')
C
C
C**** OPEN FILE 2, THE SVPRIME FILE TO BE WRITTEN TO THE NEXT LEVEL
      WRITE (FNAME,20) DATADIR,IYEAR,LEVELNM
   20 FORMAT (A,'/USVPRIME.',I2,I1)
      OPEN (2,FILE=FNAME)
C
C
C**** OPEN FILE 3, THE SORTED SVPRIME FILE (FROM PREVIOUS RUN)
      LM1 = LEVELNM - 1
      WRITE (FNAME,30) DATADIR,IYEAR,LM1
   30 FORMAT (A,'/SSVPRIME.',I2,I1)
      OPEN (3,FILE=FNAME)
C
C
C**** OPEN FILE 7, THE OUTPUT SHOPPING LIST
      WRITE (FNAME,35) DATADIR,IYEAR,LEVELNM
   35 FORMAT (A,'/SHOPLIST.',I2,I1)
      OPEN (7,FILE=FNAME,FORM='UNFORMATTED')
C
C
C**** OPEN TCOST FILE FROM THE PREVIOUS LEVELS RUN AND READ
      WRITE (FNAME,40) DATADIR,IYEAR,LM1
```

```
   40 FORMAT (A,'/TCOST.',I2,I1)
      OPEN (8,FILE=FNAME,FORM='UNFORMATTED')
      READ(8)TCOST
      IF(DEBUG)WRITE(*,*)' TCOST IN =',TCOST
C
C
C**** OPEN TCOST FILE TO BE WRITTEN IN THIS RUN
      WRITE (FNAME,40) DATADIR,IYEAR,LEVELNM
      OPEN (9,FILE=FNAME,FORM='UNFORMATTED')
C
C
C
C*****************************************************************
C** BEGIN NEW COMPONENT, READ COMPONENT RECORD FROM RESULTS FILE **
C*****************************************************************
C
   50 READ(1,END=999)NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,COMNEG,
     & COMPIP,EBOS,SVPSTART,NSONSNT,NHANSN,NSV,IBUDCODE
   60 FORMAT(A13,1X,F10.2,1X,I8,3(1X,E13.6)/4(E13.6,1X),I4,1X,A13/
     &        I5,1X,I3)
      NREAD=NREAD+1
C
      IF(DEBUG)WRITE(*,*)'  TAPE 1 HEADER   ',
     & NSN,COST,ITASSE,COMINS,COMSRU,COMNAS,COMNEG,
     & COMPIP,EBOS,SVPSTART,NSONSNT,NHANSN,NSV,IBUDCODE
C
      IF(NSONSNT.GT.0)THEN
         READ(1)(SONSNT(I),I=1,NSONSNT)
   70    FORMAT(4(1X,A13))
         IF(DEBUG)WRITE(*,*)' SONSNT ',(SONSNT(I),I=1,NSONSNT)
      END IF
      SUNKC=COMINS+COMNAS+COMNEG+COMPIP
      NPROC=SUNKC/COST+.5
      IF(DEBUG)WRITE(*,*)' NPROC,ITASSE=',NPROC,ITASSE
      MINSV=1000.
      SVPBAWT=SVPSTART
C
C
C**** READ SVPRIME FILE FROM LEVEL ABOVE FOR NSN MATCH.
  100 IF(NSNIN.GT.NSN)GO TO 110
      IF(NSNIN.EQ.NSN)THEN
C        ==== NSN'S MATCH.  TEST MINSV AND RESET IF NECC.
         IF(SVBAWT.LT.MINSV)MINSV=SVBAWT
      END IF
      READ(3,102,END=105)NSNIN,SVBAWT
  102 FORMAT(A,E14.7)
      IF(DEBUG)WRITE(*,*)'  SVBAWT READ ',NSNIN,SVBAWT
      GO TO 100
C
  105 NSNIN='99999'
C
C**** PROCESS COMPONENT, BUYING DOWN THE MARGINAL RECORDS
```

```
C****  UNTIL MINSV IS HIT
  110 LRUSBAWT=0
      IF(NSV.LE.0)GO TO 150
      DO 120 I=1,NSV
            READ(1)SV,GLCOST,NLRUS,SVP
            IF(DEBUG)WRITE(*,*)'  REC IS ',SV,GLBOST,NLRUS,SVP
            IF(SV.LT.MINSV)GO TO 130
            SVPBAWT=SVP
            LRUSBAWT=NLRUS
  120 CONTINUE
      GO TO 150
C
  130 IF(I.EQ.NSV)GO TO 150
      DO 140 J=1+I,NSV
         READ(1)SV
  140 CONTINUE
C
C
C**** COMPUTE FINAL VALUES AND WRITE
  150 IF(NSONSNT.GT.0)THEN
         DO 155 I=1,NSONSNT
            WRITE(2,102)SONSNT(I),SVPBAWT
  155    CONTINUE
      END IF
      ITARGET=ITASSE+LRUSBAWT
      NLRUSPRO=NPROC+LRUSBAWT
      IF(NLRUSPRO.EQ.0)GO TO 50
      IF(DEBUG)WRITE(*,*)'  ITARGET,NLRUSPRO=',ITARGET,NLRUSPRO
      TCOST=TCOST+COST*NLRUSPRO
      NWRITES=NWRITES+1
      IF(DEBUG)
     &   WRITE(*,160)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
  160 FORMAT(1X,A13,F10.2,4I7)
      WRITE(7)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
      GO TO 50
C
C
C*****************************************************************
C**********    WRAP UP.  WRITE TOTALS TO COST FILE     ***********
C*****************************************************************
  999 WRITE(9)TCOST
      IF (DEBUG) WRITE(*,*)'  NWRITES=',NWRITES,'  NREAD =',NREAD
      WRITE(*,*)' TOTAL EXPENDITURE FOR ALL COMPONENTS =',TCOST
      WRITE(2,102)'~~~~~~~~~~~~~~~~~~',0.
      WRITE(*,*)' '
      END
```

```
C**** SHOPB,  SHOPPING LIST PROGRAM FOR ALL COMPONENTS
C**** SHOP GENERATES SHOPPING LISTS BY BUDGET CODE AND
C**** TOTAL PURCHASES
C
      IMPLICIT INTEGER(Z)
C
      CHARACTER NSN*13,FNAME*30,TEST*6
      CHARACTER DATADIR*7,COMMENT*80,WSNAME*13
      PARAMETER (ZSUNK=10)
      REAL PBUYA(5),STCOST(ZSUNK)
      LOGICAL DEBUG
C
C
      DEBUG=.FALSE.
      IF(DEBUG)WRITE(*,*)'  OPEN LEVELNM     '
C
C**** READ DIRECTORY NAME & YEAR FROM LEVELNM FILE
      OPEN (3,FILE='LEVELNM')
      READ (3,*) IYEAR,LEVELNM,DATADIR
C
      WRITE (FNAME,6) DATADIR,IYEAR
    6 FORMAT (A,'/WSNATURE.',I2)
      OPEN (4,FILE=FNAME)
      READ (4,*) VMCONST, PBUYA
      READ (4,8) WSNAME,IFLYHRS,NUNITS,NBASES,COMMENT
    8 FORMAT(2X,A,I7,2I4,//A)
      TEST = COMMENT
C
C
C**** OPEN FILES 1 AND 2, THE SHOPLIST FILES FOR BOTH LEVELS
      DO 100 I=1,2
         IF(DEBUG)WRITE(*,*)'  OPEN SHOPLIST.',IYEAR,I
         WRITE (FNAME,10) DATADIR,IYEAR,I
   10    FORMAT (A,'/SHOPLIST.',I2,I1)
         OPEN (I,FILE=FNAME,FORM='UNFORMATTED')
  100 CONTINUE
C
C
C**** OPEN FILES 7 TO 16, THE OUTPUT SHOPPING LIST PER IBUDCODE
      DO 200 I=7,16
         IF(DEBUG)WRITE(*,*)'  OPEN SHOPBC',I-6
         WRITE (FNAME,25) DATADIR,I-6,IYEAR
   25    FORMAT (A,'/SHOPBC',I2.2,'.',I2)
         OPEN (I,FILE=FNAME)
         IF(TEST.EQ.'  NONE')COMMENT=' '
         WRITE (I,37)WSNAME,IYEAR,COMMENT,I-6
   37    FORMAT('  FOR THE "',A13,'"  SYSTEM, FOR THE YEAR 19',I2,
     &'   ',/,A80,/,
     &'                  BUDGET CODE ',I2,/,
     &'                                         NUMBER',/,
     &' COMPONENT NAME          COST    TARGET BOUGHT  LEVEL')
  200 CONTINUE
```

```
C
C
C**** OPEN FILE 18, THE OUTPUT SHOPPING LIST FOR ALL COMPONENTS
      IF(DEBUG)WRITE(*,*)'  OPEN SHOPLIST.',IYEAR
      WRITE (FNAME,26) DATADIR,IYEAR
   26 FORMAT (A,'/SHOPLIST.',I2)
      OPEN (18,FILE=FNAME)
      IF(TEST.EQ.'  NONE')COMMENT=' '
      WRITE (18,38)WSNAME,IYEAR,COMMENT
   38    FORMAT('  FOR THE "',A13,'"  SYSTEM, FOR THE YEAR 19',I2,
     &'    ',/,A80,/,
     &'                                   NUMBER         BUDGET',/,
     &' COMPONENT NAME         COST    TARGET BOUGHT  LEVEL   CODE')
C
C
C*********************************************************************
C****             READ COMPONENT RECORDS FROM SHOP FILES AND          ****
C****                  WRITE THEM TO THE BUDGET FILES                 ****
C*********************************************************************
C
      DO 300 I=1,2
   50    READ(I,END=260)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
         WRITE(IBUDCODE+6,160)NSN,COST,ITARGET,NLRUSPRO,LEVELNM
         WRITE(18,160)NSN,COST,ITARGET,NLRUSPRO,LEVELNM,IBUDCODE
  160    FORMAT(1X,A13,5X,F10.2,4I7)
         STCOST(IBUDCODE)=STCOST(IBUDCODE)+COST*NLRUSPRO
         GOTO 50
  260    CLOSE(I,STATUS='DELETE')
  300 CONTINUE
C
C
C**** WRITE SUBTOTAL COST FOR BUDGET CODE
      DO 350 I=1,ZSUNK
         WRITE(I+6,340)I,STCOST(I)
  340    FORMAT(//,' SUBTOTAL COST FOR BUDGET CODE ',I2,' =',F12.2)
  350 CONTINUE
C
C
C**** OPEN FILE 19, THE TOTAL COST FOR LEVEL "I" COMPONENTS
      DO 400 I=1,2
         IF(DEBUG)WRITE(*,*)'  OPEN TCOST.',IYEAR,I
         WRITE(FNAME,420)DATADIR,IYEAR,I
  420    FORMAT(A,'/TCOST.',I2,I1)
         OPEN(19,FILE=FNAME,FORM='UNFORMATTED')
         READ(19)TCOST
         WRITE(18,*)' '
         IF (I.EQ.1)THEN
            WRITE(18,*)' '
            WRITE(18,430)TCOST
  430       FORMAT(1X,'TOTAL EXPENDITURES FOR LEVEL 1 = ',F12.2)
         ELSE
            WRITE(18,435)TCOST
```

```
  435        FORMAT(1X,'TOTAL EXPENDITURES FOR',
     &                                  ' ALL COMPONENTS = ',F12.2)
         END IF
         CLOSE(19,STATUS='DELETE')
  400 CONTINUE
C
      WRITE(*,*)' SHOPPING DONE'
      END
```

```
C**** SPLITSRT, SORT ROUTINE FOR SORTING SAM OUTPUT
      CHARACTER SONSN(2)*13,NSNSRU(2)*13,NSNHOLD*13
      CHARACTER FNAME*30,WRK*50,DATADIR*7
      REAL GLCSRU(2),SVSRU(2)
      INTEGER IBUDSRU(2)
C
C**** OPEN THE LEVELNM FILE AND READ IN THE YEAR, LEVEL & DIRECTORY.
      OPEN(13,FILE='LEVELNM')
      READ(13,*) IYEAR,LEVELNM,DATADIR
C
C
C********************************************************************
C**    BEGIN BY SPLITTING UNSORTED.YYL INTO FILES TO BE MERGED     **
C********************************************************************
C
C**** OPEN FILE 2, THE FILE TO BE SORTED (UNSORTED.YYL)
      WRITE(FNAME,20)DATADIR,IYEAR,LEVELNM
   20 FORMAT(A,'/UNSORTED.',I2,I1)
      OPEN(2,FILE=FNAME,FORM='UNFORMATTED')
C
C**** READ FIRST RECORD AND START FIRST OUTPUT FILE.
C**** JOUT IS THE "WAVE" OF PAIRWISE MERGES.  THIS IS WAVE 0.
C**** IOUT IS THE FILE INDEX WITHIN A WAVE.
      READ(2)SONSN(2),SVSRU(2),GLCSRU(2),NSNHOLD,IBUDSRU(2)
      JOUT=0
      IOUT=1
      WRITE(FNAME,50)DATADIR,JOUT,IOUT
   50 FORMAT(A,'/SORTMP',I2.2,'.',I3.3)
      OPEN(3,FILE=FNAME,FORM='UNFORMATTED')
      WRITE(3)SONSN(2),SVSRU(2),GLCSRU(2),NSNHOLD,IBUDSRU(2)
C
C**** READ NEXT RECORD.  IF FLAGGED, CHECK NSNSRU VS NSNHOLD AND IF
C****   .LE. END THIS OUTPUT FILE AND START A NEW ONE.
  100 READ(2,END=199)SONSN(2),SVSRU(2),GLCSRU(2),NSNSRU(2),IBUDSRU(2)
      IF(SVSRU(2).GT.1.)THEN
         IF(NSNSRU(2).GT.NSNHOLD)THEN
            NSNHOLD=NSNSRU(2)
         ELSE
C        ==== START NEW FILE.  CLOSE OLD FILE 3 AND OPEN NEW ONE.
            CLOSE(3)
            IOUT=IOUT+1
            WRITE(FNAME,50)DATADIR,JOUT,IOUT
            OPEN(3,FILE=FNAME,FORM='UNFORMATTED')
         END IF
      END IF
      WRITE(3)SONSN(2),SVSRU(2),GLCSRU(2),NSNSRU(2),IBUDSRU(2)
      GO TO 100
C
C**** DONE
  199 CLOSE(2,STATUS='DELETE')
      CLOSE(3)
      WRITE(*,*)' SPLIT COMPLETED WITH',IOUT,'  OUTPUT FILES.'
```

```
C
C
C
C
C
C*****************************************************************
C**  OOOO  NOW MERGE FILES, TWO AT A TIME, INTO SORTED.YYL  OOOO  **
C*****************************************************************
C************* WAVE LOOP.  N WAVES CAN MERGE 2^n FILES. ***********
C**** REDUCE NUMBER OF FILES BY A FACTOR OF 2 BY MERGING PAIRS.
C****    EACH MERGE IS A NEW WAVE.  JOUT IS WAVE INDEX.  IOUT IS FILE
C****    INDEX.  IOUTLAST IS MAXIMUM IOUT FOR LAST WAVE.  IOLD IS FILE
C****    INDEX FOR PREVIOUS WAVE.  IF IOUTLAST=1, LAST WAVE HAD ONLY
C****    ONE FILE (WHICH IS SORTED), ERGO YOU'RE DONE.
  200 IOUTLAST=IOUT
      IF(IOUTLAST.EQ.1)GO TO 900
      JOLD=JOUT
      JOUT=JOUT+1
      IOUT=1
      IOLD=1
C
C
C
C     ======= IOLD/IOUT LOOP, MERGE PAIRS TIL <2 LEFT.  ===========
C     ==== MERGE TWO FILES (JOLD,IOLD & JOLD,IOLD+1) INTO ONE
C     ====    (JOUT,IOUT).  FIRST OPEN FILES AND READ INITIAL RECORDS.
  300    WRITE(FNAME,50)DATADIR,JOLD,IOLD
         OPEN(1,FILE=FNAME,FORM='UNFORMATTED')
         READ(1)SONSN(1),SVSRU(1),GLCSRU(1),NSNSRU(1),IBUDSRU(1)
C
  320    WRITE(FNAME,50)DATADIR,JOLD,IOLD+1
         OPEN(2,FILE=FNAME,FORM='UNFORMATTED')
         READ(2)SONSN(2),SVSRU(2),GLCSRU(2),NSNSRU(2),IBUDSRU(2)
C
         WRITE(FNAME,50)DATADIR,JOUT,IOUT
         OPEN(3,FILE=FNAME,FORM='UNFORMATTED')
C
C
C        ========  MERGE LOOP.  MERGE TWO SORTED STACKS.  =========
C        ==== MERGE ONE RECORD.  COMPARE AND PROCESS WINNER.
  400    IF(NSNSRU(1).LT.NSNSRU(2))THEN
               IWON=1
            ELSEIF(NSNSRU(1).EQ.NSNSRU(2).AND.SVSRU(1).GT.SVSRU(2))THEN
               IWON=1
            ELSE
               IWON=2
         END IF
         WRITE(3)SONSN(IWON),SVSRU(IWON)
     &         ,GLCSRU(IWON),NSNSRU(IWON),IBUDSRU(IWON)
         READ(IWON,END=499)SONSN(IWON),SVSRU(IWON)
     &           ,GLCSRU(IWON),NSNSRU(IWON),IBUDSRU(IWON)
         GO TO 400
```

```
C
  499       NSNSRU(IWON)='~~~~~~~~~~~~~~~~~~'
            IF(NSNSRU(1).NE.NSNSRU(2))GO TO 400
C        ==================== END OF MERGE LOOP ==================
C
C
C     ==== THIS MERGE COMPLETE.  TIME TO MERGE NEXT TWO.
C     ====   CLOSE AND ERASE OLD FILES.
  500    CLOSE(1,STATUS='DELETE')
         CLOSE(2,STATUS='DELETE')
         CLOSE(3)
C
C     ==== IF IOLD NOT NEAR IOUTLAST, KEEP GOING,
         IF(IOLD.LT.IOUTLAST-2)THEN
            IOLD=IOLD+2
            IOUT=IOUT+1
            GO TO 300
         END IF
C     ==================== END OF IOLD LOOP ====================
C
C
C
C**** ELSE GO TO NEXT WAVE.
      IF(IOLD.EQ.IOUTLAST-2)THEN
C     ==== SPECIAL CASE FOR ODD #ed LAST FILE.  JUST COPY TO NEW WAVE.
         IOUT=IOUT+1
         WRITE(WRK,550)DATADIR,JOLD,IOUTLAST,DATADIR,JOUT,IOUT
  550    FORMAT('mv ',A,'/SORTMP',I2.2,'.',I3.3,1X,A,'/SORTMP'
     &         ,I2.2,'.',I3.3)
         CALL SYSTEM(WRK)
      END IF
      GO TO 200
C*********************** END OF WAVE LOOP ***********************
C
C
C
C
C*****************************************************************
C**  000000000  NOW JUST MOVE LAST FILE TO SORTED.YYL  000000000  **
C*****************************************************************
  900 WRITE(WRK,950)DATADIR,JOUT,IOUT,DATADIR,IYEAR,LEVELNM
  950 FORMAT('mv ',A,'/SORTMP',I2.2,'.',I3.3,1X,A,'/SORTED.'
     &         ,I2,I1)
      CALL SYSTEM(WRK)
      WRITE(*,*)' SPLITSRT COMPLETED SUCCESSFULLY'
      END
```

```
C**** SRUSTART,  STARTS FILLING THE SRUEBO & SVPRIME ARRAYS
      SUBROUTINE SRUSTART
C
C**** THIS SUBROUTINE INITIALIZES THE SRUEBO (& CO.) ARRAYS.
C**** IF NSRUEBO IS RETURNED AS 0. THEN NO CHILDREN.
C**** THIS SUBROUTINE ASSUMES THAT NSN,NSNSRU,SORTV,GLCSRU,
C****  & SONSN ARE DEFINED.
C
      IMPLICIT INTEGER(Z)
C
      COMMON/GENCHAR/NSN
      COMMON/GENERAL/DEBUG,Q,COST
      CHARACTER NSN*13
      LOGICAL DEBUG
C
      COMMON/SRUCHA/NSNSRU,SONSN,SONSNT
      COMMON/SRUBLK/SRUEBO,SVPRIME,NSRUEBO,ISRUSTRT
     & ,SVSRU,GLCSRU,RGLCSRU,NSONSNT,NSREADS
      PARAMETER (ZSRUEBO=2000,ZSONSNT=300)
      REAL SRUEBO(0:ZSRUEBO),SVPRIME(0:ZSRUEBO)
      CHARACTER SONSN*13,SONSNT(ZSONSNT)*13,NSNSRU*13
C
C**** INITIALIZE.
      NSRUEBO=0
      SRUEBO(0)=0.
      NSONSNT=0
C
C**** PROCESS RECORD UNLESS PAST NSN.
      IF (DEBUG) WRITE (*,10) NSNSRU,NSN
   10 FORMAT (' NSNSRU = ',A,' NSN = ',A)
  100 IF(NSNSRU.GT.NSN)RETURN
C
C**** IF A MATCH PROCESS ALL FLAGGED RECORDS.
C     ***********************************************************
C
      IF(NSNSRU.EQ.NSN)THEN
C
C        ==== IF A REAL SVSRU IS ENCOUNTERED, YOU'RE DONE.
         IF(SVSRU.LE.500.)RETURN
C
C        ==== SUPER LARGE SVSRU IS A FLAG THAT THIS RECORD IS
C        ====   REALLY A STARTING RECORD.  SAVE.  THE GLCSRU
C        ====   IN THESE RECORDS IS ACTUALLY STARTING EBO.  SUM.
C
         SRUEBO(0)=SRUEBO(0)+GLCSRU
         SRUEBO(1)=SRUEBO(0)
         NSONSNT=NSONSNT+1
         NSRUEBO=1
         IF(NSONSNT.GT.ZSONSNT)STOP ' NSONSNT > Z'
         SONSNT(NSONSNT)=SONSN
      END IF
C
```

```
C     ******************************************************
C
C**** READ NEXT RECORD AND LOOP BACK.
C
      READ(3,END=199)SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
  150 FORMAT (' ',A13,1X,E13.7,1X,F13.2,1X,A13,I3)
      IF (DEBUG) WRITE (*,150) SONSN,SVSRU,GLCSRU,NSNSRU,IBUDSRU
      NSREADS=NSREADS+1
      GO TO 100
C
C
C**** EOF.  SET HIGH NSNSRU TO PREVENT FURTHER READS.
  199 NSNSRU='~~~~~~~~~~~~~~~~~~'
C
      IF(DEBUG)WRITE(*,*)'--> IN SRUSTART NSONSNT[OUT]= ',NSONSNT
C
      RETURN
      END
```

```
C**** UPDATA,  UPDATES THE COMPONENT DATA FILES BASED ON
C               FLYING HOUR PROGRAM
      SUBROUTINE UPDATA(DATADIR,IYEAR,FH)
C
      CHARACTER DATADIR*7,FNAME*30,NSN*13,NHA*13
      INTEGER ASSET,BUDCOD
C
C**** READ AIRCRAFT TYPE BASELINE FLYING HOUR PROGRAM
      WRITE(FNAME,10)DATADIR
   10 FORMAT(A,'/HOURS')
      OPEN(3,FILE=FNAME,ERR=900,STATUS='OLD')
      READ(3,*)BASHRS
C
C**** CHECK FOR OLD DATA FILES AND ADJUST PIPELINES.
      DO 500 I=1,2
         WRITE(FNAME,20)DATADIR,IYEAR,I
   20    FORMAT(A,'/DATA.',I2,I1)
         OPEN(1,FILE=FNAME,ERR=900,STATUS='OLD')
         WRITE(FNAME,30)DATADIR,IYEAR,I
   30    FORMAT(A,'/COMPDATA.',I2,I1)
         OPEN(2,FILE=FNAME,FORM='UNFORMATTED')
  100    READ(1,*,END=200)NSN,COST,BR,OS,DR,CR,ASSET,NHA
     &                   ,LEVEL,BUDCOD
         BR=FH*BR/BASHRS
         OS=FH*OS/BASHRS
         DR=FH*DR/BASHRS
         CR=FH*CR/BASHRS
         WRITE(2)NSN,COST,BR,OS,DR,CR,ASSET,NHA,LEVEL,BUDCOD
         GOTO 100
  200    CLOSE(1)
         CLOSE(2)
  500 CONTINUE
      GOTO 999
C
  900 WRITE (*,*) ' DATA FILES NOT FOUND.  ',
     &           'PLEASE CHECK DIRECTORY "',DATADIR,'"'
      CALL GETPID(ID)
      CALL KILL(ID,9)
C
  999 CONTINUE
      RETURN
      END
```

Bibliography

1. Abell, John B. "Resources to Readiness: The Tactical Fighter Case," Defense Management Journal, 18 : 15-19 (Spring Quarter 1981).

2. AFLCP 57-13. Recoverable Inventory Control Using MOD-METRIC. Wright-Patterson Air Force Base: Head-quarters Air Force Logistics Command, Department of the Air Force, February 1975.

3. Blom, Raymond and John H. Evans III "Forecasting Air Force Logistics Manpower Requirements," Defense Management Journal, 17 : 18-23 (Fourth Quarter 1981).

4. Brabson, G. Dana "Department of Defense Acquisition Improvement Program," Concepts, 4 : 54-75 (Autumn Quarter 1981).

5. Einbu, John M. "Allocation of Resources to Depots," Journal of Operations Research Society, 36: 929-933 (October 1985).

6. Feeney, George J. and Craig C. Sherbrooke. " The (s-1,s) Inventory Policy Under Compound Poisson Demand," Management Science, 12 (5): 399-411 (January 1966).

7. Holcomb, M. S. Relating Resources to Readiness Study. CNA Report 1, Chief of Naval Operation, (September 1979).

8. Horowitz, Sanley A. "Quantifying Seapower Readiness," Defense Management Journal, 18: 3-7 (Spring Quarter 1981).

9. Monahan, Richard H. An Evaluation Model Concept For Relating Task Group Readiness to Logistic Support System Performance: Final Report, Contract No. N00167-79-0-0197, SRI Project 8976, SRI International, Menlo Park, California, (June 1981).

10. Muckstadt, John A. "A Model for Multi-Item, Multi-Echalon, Multi-Indenture Inventory System," Management Science, 20 (4): 472-481 (December 1973).

11. O'Malley, T. J. The Aircraft Availability Model: Conceptual Framework and Mathematics, Logistics Management Institute, June 1983.

12. Palm, C. " Analysis of the Erlang Traffic Formula for Busy-Signal Arrangemnets," Ericson Technique No. 5, 39-45, 1938.

13. Pristker, A. Alan B. Introduction to Simulation and SLAM II, (Second Edition). New York: Jhon Willey & Sons, Inc., 1984.

14. Puritano, Vincent and Paul J. Berenson. Improving the Defense Acquisition System and Reducing System Costs. Department of Defense Steering Group Report, (March 1981).

15. Radivoj Petrovic, Aleksander Senbonn and Minko Vujosevic "Spares Allocation in the Presence of Uncer tainty," European Journal of Operation Research, 11: 77-81, Mihailo Pupin Institute, Belgrade, Yugoslavia, (September 1982).

16. RFA 45-3. Aircraft Readiness Analysis Program. Headquarters Portuguese Air Force, January 1974.

17. RFA 415-1. Air Force Supply Manual. Head-quarters Portuguese Air Force, September 1980. 10-1,10-2.

18. Rippy Douglas V. and Walston, Steven W. Evaluation of the USAF/Logistics Management Institute FXDP Model. AFLC Report PPBR Task Order 77-6, September 1978.

19. Stokes, Major Ronald. Aircraft Availability Model User's Manual. HQ AFLC/XRS, Wright Patterson AFB, June 1986.

20. Sherbrooke, Craig C. "METRIC: A Multi-Echelon Technique for Recoverable Item Control," Operations Research, 16: 122-141 (1968).

21. -----. Discrete Compound Poisson Processes and Tables of the Geometric Poisson Distribution. RAND Corporation, July 1966.

22. Sherif, Y. S. and N. A. Khein. Weapons Systems Analysis: Technical Report, RD-OR-82-2, The University of Alabama in Huntsville, Huntsville, Alabama (October 1981).

23. Talhinhas, Major C. M. P. Formulation of a Mathemetical Model to Study the Portuguese Air Force Flying Rate in Terms of the Available Human and Material Resources. Student Project Report, Air Force Staff, 3rd Division, Portuguese Air Force, 1984.

VITA

Major Rui Manuel Pereira Lopes was born on 17 july 1950 in Mafra, Lisbon, Portugal. He graduated from Military Academy in June 1976 with the degree of Engineer in Military Aeronautical Engineering. Upon graduation he was assigned to the industrial depot, Oficinas Gerais de Material Aeronautico, until 1979. He completed the Air Staff College Program in 1980. He then served as an weapon system manager in the Directorate of Maintenance and Supply for aeronautical and mechanical systems, Direccao de Servico de Material. In 1983, when the Portuguese Air Force Logistics Command was created, he served as the A-7P manager officer under the Directorate of Mechanical and Aeronautical, Direccao de Mecanica e Aeronautica, until entering the School of Engineering, Air Force Institute of Technology, in May 1985.

Permanent address: R. DR. JOAO COUTO 6, 8 B
1500 LISBON
PORTUGAL

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

A179 517

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release Distribution Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GOR/OS/86D-8	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering AF Institute of Technology 6b. OFFICE SYMBOL (If applicable) AFIT/ENS	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code) Wright-Patterson AFB Ohio, 45433-6583	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification) See Box 19	

12. PERSONAL AUTHOR(S): Lopes, Rui Manuel Pereira

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
MS Thesis	FROM ______ TO ______	1986 December	161

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	
5	9		
15	5		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title: AIRCRAFT READINESS UNDER BUDGET CONSTRAINT, ITS RELATIONSHIP WITH LOGISTICS RESOURCES; THE SPARE PARTS COMPONENT, A MODEL FOR THE PORTUGUESE AIR FORCE

Thesis Advisor: Joseph R. Litko, Ph.D., Major, USAF
Assistant Professor, Department of Operational Sciences

Approved for public release: IAW AFR 190-1.
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AFC)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Joseph R. Litko, Major, USAF	22b. TELEPHONE NUMBER (Include Area Code) (513) 255-3362 22c. OFFICE SYMBOL AFIT/ENS

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

One of the general concerns of the Portuguese Air Force top managers is to relate resources to readiness. This research addresses the relationship between spare parts and aircraft availability, as a component of the general problem of relating resources to readiness. As background, the theoretical development and problem solution techniques of METRIC: A Multi-Echelon Technique for Recoverable Item Control, MOD-METRIC: A Model for Multi-Item, Multi-Echelon, Multi-Indenture Inventory System, and AAM (Aircraft Availability Model) models are presented. After identifying the major mathematical issues and contributions of each model to the solution of the problem, an easy-to-use mathematical computer model, that simplifies the actual Aircraft Availability Model in use by the Air Force Logistics Command (AFLC) and that fits the requirements of the Portuguese Air Force, is presented. The simplified model for aircraft availability provides solutions that are very close to historic values. An additional benefit of the simplified model is that it can be used to predict either aircraft availability or total expenditures. It is recommended that weapon system managers in the Portuguese Air Force use the model to budget the spare parts requirements for each aircraft type.

DATE
FILMED
5-82